Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Irányítástechnika és Informatika Tanszék

Nem átlapolódó 2D alakzatok hatékony alaktartó interpolációja

TDK dolgozat

Készítette:

Sánta Márton

Konzulens:

Dr. Csébfalvi Balázs

2021

Tartalomjegyzék

Kivonat

A hagyományos alaktartó interpoláció (shape-based interpolation) képes folytonos átmenetet generálni akár topológiailag különböző alakzatok kontúrjai között is. A kontúrok alapján először egy-egy 2D előjeles távolságmezőt értékelünk ki, majd a távolságmezők között lineárisan interpolálunk. Az interpolált távolságmező az átmeneti kontúr implicit reprezentációja, amiből a masírozó négyzetek (marching squares) algoritmussal lehet kinyerni az átmeneti kontúr geometriai modelljét.

Ez a megközelítés azonban csak akkor működik, ha az alakzatok között van átlapolódás. A gyakorlatban ugyanakkor ezt a feltételt nem mindig lehet teljesíteni. Előfordulhat például, hogy csőszerű felületrészeket kell rekonstruálni úgy, hogy akár nem átlapolódó keresztmetszetek között is szeretnénk folytonos összeköttetést biztosítani (például érhálózat rekonstrukciója keresztmetszeti CT szeletek alapján). Ebben a dolgozatban azt vizsgáljuk, hogy miként lehet a korábbi alaktartó interpolációs módszereket kiterjeszteni nem átlapolódó alakzatok interpolációjára, illetve hogyan lehet biztosítani a távhatás paraméterezhetőségét.

Abstract

Conventional shape-based interpolation can generate a continuous transition even between contours of topologically different shapes. Based on the contours, we first calculate the 2D signed distance transform and then interpolate linearly between the distance maps. The interpolated distance map is an implicit representation of the intermediate contour, from which the geometric model of this contour can be extracted using the marching squares algorithm.

However, this approach only works if the shapes overlap. In practice, however, this condition cannot always be met. For example, it may be necessary to reconstruct tubular surface parts so as to provide a continuous connection even between non-overlapping cross-sections (e.g., reconstruction of a vascular network based on cross-sectional CT slices). In this paper, we examine how the previous shape-based interpolation methods can be extended to the interpolation of non-overlapping shapes, and how the parameterization of the remote effect can be ensured.

1. fejezet

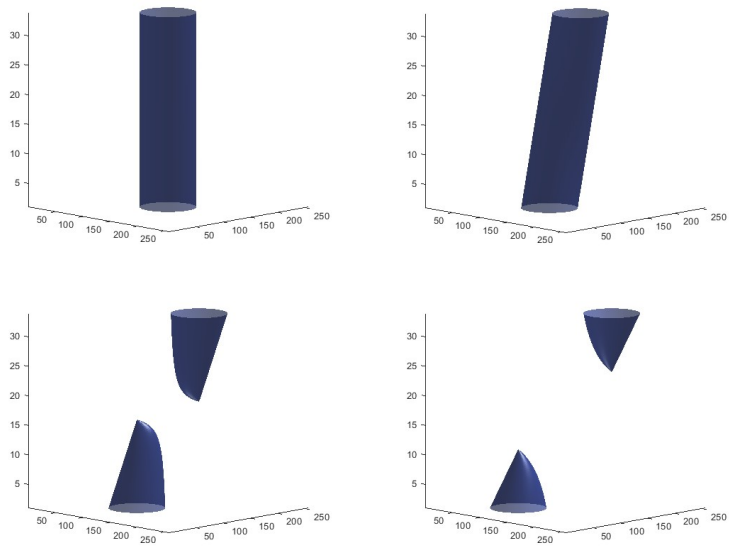
Bevezetés

A keresztmetszeti kontúrok alapján történő 3D modellezés a számítógépes grafika és a modellezés egyik kiemelten fontos és aktívan kutatott területe. Ennek számos kidolgozott eljárása található meg a szakirodalomban. Lényege, hogy rendelkezésre állnak egy objektumról készült keresztmetszeti felvételek (szeletek), amelyeken kontúrokat detektálunk. A detektálás egy bizonyos küszöbérték definiálásával, például a *marching squares* [1] algoritmussal történhet. A cél ezen keresztmetszeti kontúrok közötti folytonos átmenet megteremtése, akár köztes fázisok generálása és ezek alapján a 3D felület rekonstrukciója.

Egyik fő csoportja a lehetséges megoldásoknak az implicit függvényekkel történő megközelítés [2], amely nem közvetlenül a kontúrok geometriai modelljével operál, hanem azok egy implicit leírójával. Ez a reprezentáció sok olyan előnyt jelent a feldolgozás folyamán, amelyek magából a függvénnyel való jellemzésből következnek, és alkalmazása mellett szólnak. Erre egyik példa, hogy a topológiailag különböző régiók/kontúrok között is folytonos és az intuíciónak megfelelő átmenetet tud teremteni, abban az esetben, ha ezek a régiók/kontúrok egymással átlapolódnak. Ezzel szemben a direkt geometriai jellemzés esetén külön megoldandó feladatot jelent a kontúrpontok egymásnak történő megfeleltetése, a köztük történő interpoláció, mesh generálása a megjelenítéshez, stb.

Implicit függvények használata esetén köztes szeletek generálásakor, általában a leíró függvények között interpolálunk valamilyen módszerrel, és az eredményre, mint a köztes fázisok kontúrjainak implicit leírójára gondolunk. A számos előny mellett hátránya is van ezeknek a módszereknek. Ha egyszerű lineáris interpolációt alkal-

mazunk két fázis között, akkor a nem átlapolódó régiók/kontúrok közötti folytonos átmenet nem lehetséges, ahogyan az 1.1 ábrán is megfigyelhető. A két szelet között további szeleteket generálva, az egyiken detektált kontúr fokozatosan eltűnik, míg a másikon detektált kontúr pedig megjelenik, de a kontúrok távolságától függően több olyan köztes szelet is előfordulhat, amelyeken nem detektálható kontúr, az átmenet nem folytonos. A 3D felületeket tekintve, a két régiót a közelítés külön-külön lezárja.



1.1. ábra. Szomszédos szeleteken detektált keresztmetszeti kontúrok közötti lineáris interpoláció változó kontúrtávolság mellett. Jól megfigyelhető, hogy a távolság növekedésével, a folytonos összeköttetés egyszer csak megszűnik és a két felületelem külön-külön lezárásra kerül.

Ez hátrányt jelenthet például nagy görbületű felületek rekonstrukciója esetén, ha a mintavételek, azaz a szeletek száma ezt nem követi le. Célszerű egy olyan algoritmus kifejlesztése, amely képes ezekben az esetekben is elfogadható megoldást kínálni, felhasználói kontroll mellett. Ennek célja a távhatás szabályozása, azaz annak megszabása, hogy egymástól maximálisan milyen távolságra elhelyezkedő kontúrok között szeretnénk még folytonos átmenetet generálni és mely esetekben kívánjuk a felületelemek tényleges lezárását.

Hasznos lehet az eljárás például érhálózat CT felvételek alapján történő rekonstrukciója esetén, ha a rendelkezésre álló szeletek száma felülről korlátos, rugalmasságot és robusztusságot biztosítva ezzel, felhasználói ráhatás mellett. További motivációt jelenthet, a modell tömörítése és minél kevesebb szelettel történő, mégis pontos jellemzése, a térbeli, vagy (általánosítva a módszert) akár az időbeli felbontás növelése.

A dolgozatban tehát az implicit reprezentáció alkalmazásának egyik hátrányára keresek korábban a szakirodalomból nem ismert megoldást, amely a fentiekben vázolt esetekben képes jobban teljesíteni és elfogadható megoldást kínálni.

A második fejezetben a területhez tartozó elméleti hátteret, illetve a kapcsolódó szakirodalmat ismertetem és azok kapcsolódását a jelen dolgozathoz. A harmadik fejezetben bemutatom a feldolgozás lépéseit, és a számítógépen implementált algoritmusokat, majd bemutatok a negyedik fejezetben egy új módszert a nem átlapolódó kontúrok és felületek közötti folytonos átmenet kezelésére. A dolgozatot az eredmények összefoglalásával és a jövőbeli célok megfogalmazásával zárom az ötödik fejezetben.

2. fejezet

Szakirodalmi áttekintés

A témával, illetve az ehhez kapcsolódó elméleti háttérrel számos publikáció foglalkozik, amelyek közül a legfontosabbakat a jelen fejezetben tekintem át, bemutatva a különböző megközelítéseket és megoldásokat. Mindezek mellett azokat az elméleti alapokat is összefoglalom, amelyek a bemutatott módszer továbbfejlesztéséhez szükségesek.

2.1. Alaktartó interpoláció

Az alaktartó interpolációt alkalmazó (*shape-based interpolation*) megközelítések közül az egyik kiemelten fontos és sokszor hivatkozott munka Bors és társai nevéhez fűződik [3]. A módszerük általában bináris képeken alapszik és az ezekre számított távolság-transzformációval, illetve különböző morfológiai műveletekkel közelítik meg a feladatot.

Az említett cikkben morfológiai műveleteket alkalmaztak, a rendelkezésre álló binarizált szeletfelvételek között további szeleteket generáltak és ezáltal egy sokkal nagyobb felbontású adathalmaz képezte a pontosabb modellalkotás alapját. A bemutatott konkrét esettanulmányban fogak rekonstruálásán keresztül mutatták be az alkalmazott módszer működőképességét és hatékonyságát. Az ehhez felhasznált adatbázis egy-egy fog mechanikai felszeletelése utáni beszkennelt felvételeket tartalmazta. A képek szegmentálása révén bináris képeket állítottak elő és ezekre alkalmazták az említett morfológiai műveleteket.

Ha P és Q a két bináris objektum, amelyek között az átmenetet szeretnénk generálni, akkor P-ből a Q-ba való átmenethez tartozó szeleteket a következő művelet iteratív végrehajtásával számítjuk ki, ahol a leállási feltétel az idempotencia:

$$f(P|Q,B) = [(P \ominus B) \cup ((P \cap Q) \oplus B)] \cap (P \cup Q) \tag{2.1}$$

Itt B a strukturáló elemet jelöli, a fenti kifejezés pedig azt a folytonos átmenetet fejezi ki, amely szerint P erodálása a B-vel éppen P fogyását eredményezi, míg P és Q metszetének dilattálása az átmenetet növeli B-től függő mértékben. További feltételként a műveletek a két bináris objektum unióján belül értelmezettek.

Egy másik munka, amelyben hasonlóan bináris képekre számított távolságtranszformációt alkalmaztak, Alencar Lotufo és Xavier Falcao publikációja [4]. Megközelítésükben a bináris szeletképekre egy-egy előjeles távolság-transzformációt számítottak ki, amelyet kombináltak a szürkeárnyalatos képekből kinyerhető információval. A kettő segítségével ők is köztes szeleteket generáltak, pontosítva ezzel a modelljüket.

A két módszer egyik hátránya, hogy bináris képekkel operálnak, amelyek a véges felbontásból adódóan a rekonstruált felületek egyenetlenségét fogják eredményezni. A rekonstrukciót a meglévő szeletek számának növelésével végzik el. A felületek megjelenítésénél jól megfigyelhetőek a pixelességből adódó artifaktumok.

Egy másik hátrány, hogy a bemutatott módszerekkel a közbenső szeletek generálása során, a modell előállításakor nem vesznek figyelembe egyéb, például a simaságra vonatkozó megkötéseket, mint ahogyan teszi azt a következő részben ismertetett módszer.

2.2. Implicit függvényekkel történő megközelítés

A másik technika implicit függvények használatán alapul. Ebben az esetben a keresztmetszeti kontúrokat egy-egy implicit függvény reprezentálja 2D-ben, amelynek maghatározott értékhez tartozó szintvonalai jelentik a görbék geometriai modelljét. Turk és O'Brien is ezt a megközelítést alkalmazták [2] folytonos és minimális összgörbületű alaktartó interpolációra.

Az implicit függvény megválasztásának általános és tipikus esete az előjeles távolság térkép, amely a zárt görbe belsejében negatív, kívül pozitív értékeket (vagy akár fordítva), a görbe mentén pedig zérus értéket vesz fel. Ezek az előjeles értékek a kontúrtól való távolságot reprezentálják. Az egyszerű távolság-transzformáció eredménye nem kellően sima és általában sok helyen éles töréseket tartalmaz, ami miatt a szerzők egy hasonló módon definiált, de más módszerrel előállított implicit függvényt javasoltak.

A kontúr mentén mintapontok kijelölése után peremfeltételeket definiáltak, azaz a görbe belsejében negatív, kívül pozitív, a görbén zérus értéket írtak elő, majd ezek alapján meghatározták az implicit függvényt szórt adat interpoláció (*scattered data interpolation*) segítségével.

Az implicit függvény zérus értékhez tartozó szintvonala definiálja a görbét, amely a minimális összgörbületet megkövetelő kritériumoknak eleget tesz. A kontúrt nem 2D-ben szeretnénk egy függvénnyel reprezentálni, hanem több keresztmetszeti kontúr alapján a 3D felületet szeretnénk modellezni, ezért célszerű nem külön kezelni az egyes keresztmetszeti kontúrokat, hanem egyetlen 3D implicit függvényt definiálni a fent leírtak megfelelő általánosításával, vagyis a kényszereket eggyel magasabb dimenzióba ágyazni. Így egy magasabb dimenziós implicit függvény állítható elő, amelynek a zérus értékhez tartozó szintfelülete definiálja a végeredményt, azaz a folytonos felületreprezentációt.

2.3. Szórt adat interpoláció

A szórt adat interpoláció (*scattered data interpolation*) [5] technika lényege, hogy a térben elszórtan elhelyezkedő mintavételi pontokban ismert az interpolálni kívánt függvény értéke. A cél egy olyan függvény definiálása, amely illeszkedik a diszkrét mintákra. Erre több megoldás is kínálkozik:

- Shepard-interpoláció [6]: a köztes pontokban a függvény értékét a mintavételi pontokban ismert értékek súlyozott összegéből számítja, ahol a súlyokat a mintavételi pontoktól vett távolságok reciprokai jelentik.
- Kernel regresszió (approximál a kijelölt pontokban)

- *Moving Least Squares* módszer (approximál a kijelölt pontokban) [7]
- Lineáris interpoláció
- Radiális bázisfüggvények (*Radial Basis Function* - RBF) használata [8]

A Radiális bázisfüggvények használata a Turk és O'Brien által kidolgozott megoldásban [2] is megjelenik. Ebben az esetben a megoldást úgynevezett bázisfüggvények lineáris kombinációjaként keressük:

$$\hat{f}(x) = \sum_{k=1}^{N} w_k \phi(\|\mathbf{x} - \mathbf{x_k}\|) \tag{2.2}$$

ahol a ϕ függvény az RBF és értéke csak az egyes $\mathbf{x_k}$ mintavételi pontoktól vett távolságtól függ. A RBF megválasztásához számos szempontot lehet vizsgálni, például az interpolált függvény differenciálhatóságát, a folytonossági rendjét vagy az összgörbületének minimalizálását. Az RBF családból az úgynevezett *thin plate spline* elégíti ki 2D-ben a görbület minimalizálására vonatkozó alábbi előírást.

$$\phi(r) = r^2 \log(r) \tag{2.3}$$

$$E = \int_{\Omega} f_{xx}^2(\mathbf{x}) + f_{yy}^2(\mathbf{x}) + 2f_{xy}^2(\mathbf{x}) \tag{2.4}$$

ahol E az összgörbület, amit minimalizálni szeretnénk.

Az egyes mintavételi pontokban előírt értékekre egy-egy egyenlet írható fel a fentiek alapján, melyek egy lineáris egyenletrendszerré állnak össze. Ennek megoldásával az előírt, és korábban bemutatott kényszerfeltételek kielégíthetőek.

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \cdots \\ \phi_{21} & \phi_{22} & \phi_{23} & \cdots \\ \phi_{31} & \phi_{32} & \phi_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \end{bmatrix} \tag{2.5}$$

$$\phi_{ij} = \phi(\|\mathbf{x_i} - \mathbf{x_j}\|) \tag{2.6}$$

$$f_i = f(\mathbf{x_i}) \tag{2.7}$$

Dolgozatomban szintén az implicit függvénnyel történő megközelítést követem, de az implicit függvény az egzakt Euklideszi előjeles távolság-transzformált, nem pedig a Turk és O'Brien által javasolt függvény. Ennek oka, hogy a kontrollpontok automatikus kiválasztása nem egyszerű, így az RBF reprezentáció kiszámítása sem az. További kihívást jelenthet, ha a kontrollpontok egymáshoz közel esnek és ezáltal a kényszereket definiáló egyenletrendszerben a mátrix rosszul kondicionálttá válik, ami numerikus instabilitáshoz vezethet. További nehézség, hogy a minél pontosabb kontúr reprezentációhoz sok kontrollpontra, és ezáltal peremfeltételre van szükség, amely a megoldásnál használt mátrix méretét jelentősen növeli, a számításokat bonyolítja. A minimális összgörbületre való törekvés pedig az éleket, sarkokat és ezáltal a részleteket mossa el.

A távolság-transzformáció alkalmazásával, viszont egy kényelmes és robusztus reprezentációt kapunk. Az egyik legfőbb előnye, hogy nem kell a detektált kontúrpontok közötti pontonkénti megfeleltetéssel foglalkozni. Egyes megközelítések ugyanis a modellt úgy próbálják megalkotni, hogy a szomszédos kontúrok pontjainak megfelelő párosítása segítségével definiálnak a megjelenítéshez szükséges háromszög elemeket. Ez a fajta megközelítés nem képes különböző topológiájú elemeket egyszerűen kezelni, hiszen a modellezés szempontjából kritikus a megfeleltetés eredménye és így a folytonos átmenet nem minden esetben biztosított. Ezzel foglalkozott Meyers és Skinner [9] is. Az implicit reprezentáció kiküszöböli a különböző topológiájú esetek megkülönböztetését és külön kezelését. Az ezek közötti folytonos és szép átmenet a reprezentációváltásból adódik.

További előny, hogy ezzel a jellemzéssel bizonyos műveletek sokkal könnyebben elvégezhetőek, mint például az unió képzés több alakzat esetén, vagy az alakzatok megvastagítása stb..

2.4. Optical flow alapú megközelítés

Az optical flow [10] képsorozatokon észlelhető elmozdulások detektálására alkalmas eljárás. Ahogyan az objektumok a 3D térben mozognak, úgy az egyes pontjaik vetülete a 2D képsíkra egy-egy trajektóriát ír le. Az optical flow feltevése az, hogy ezen trajektória mentén az egyes pontok (a képen pixelek) intenzitása nem változik,

állandó marad, ebből kiindulva írható fel az optikai áramlás alapegyenlete, amelyből egy elsőrendű közelítés származtatható az alábbiak szerint:

$$I_1(x, y, t) = I_2(x + \Delta x, y + \Delta y, t + \Delta t) \tag{2.8}$$

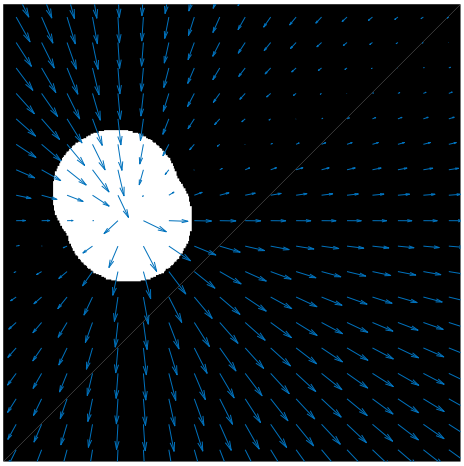
$$I_2(x + \Delta x, y + \Delta y, t + \Delta t) = I_1(x, y, t) + \frac{\partial I}{\partial x} \cdot \Delta x + \frac{\partial I}{\partial y} \cdot \Delta y + \frac{\partial I}{\partial t} \cdot \Delta t \tag{2.9}$$

$$I_2(x + \Delta x, y + \Delta y, t + \Delta t) = I_1(x, y, t) + \frac{\partial I}{\partial x} \cdot u + \frac{\partial I}{\partial y} \cdot v + \frac{\partial I}{\partial t} \tag{2.10}$$

amelyben $I(.)$ az (kép)intenzitást jelöli, x és y a pont vetületének (pixel)koordinátái, t az idő, u és v pedig a becsülendő áramlás x és y irányú komponensei, amelyek minden pixel pozícióban a trajektória menti pillanatnyi elmozdulást írják le.

Az egyenlet alulhatározott 2D-ben így az egyértelmű megoldáshoz további megkötésekre is szükség van, mint például a simaságra, minimális divergenciára [11] stb. A számításokra számos kidolgozott algoritmus létezik, mint például a Lucas-Kanade [12] vagy a Horn–Schunck [13] módszer. Az optikai áramlás számításához általában valamilyen iteratív eljárás szükséges, amelyek fokozatosan pontosítják az eredményt, amely a kényszereknek minél jobban eleget tesz.

Kis elmozdulásokra (néhány pixel) ezek az eljárások kiválóan alkalmazhatóak (lásd a 2.1 ábrát), azonban a nagy elmozdulások problémákhoz vezethetnek, hiszen az elsőrendű közelítés ebben az esetben már nem állja meg a helyét. A feladat ekkor is megoldható például képpiramisok alkalmazásával, minden piramis szinten meghagyva az elsőrendű közelítéskor használt eljárást. Ilyen algoritmus például a durva-finom iteratív Lucas-Kanade módszer [14]. Ha a képek felbontását lecsökkentjük, akkor a pixeltérben mért elmozdulások is lecsökkenek. Ha egy-egy olyan képpiramist építünk fel, amelyek egymást követő szintjein a felbontás mindig felére csökken, úgy a piramisok csúcsán lévő képpár között az elmozdulás meghatározására egyszerűen alkalmazható az alapmódszer. A piramisban lefelé haladva, az előző szinten keletkezett eredményt és az azzal újramintavételezett képet egyszerűen skálázni kell a felbontás változásának megfelelően, majd ehhez hozzá kell adni az adott szinten számított inkrementális eredményt és haladni kell a következő szint felé.



2.1. ábra. Az *optical flow* szemléltetése egy egyszerű esetben, kis elmozdulásokkal.

A módszer egyik alkalmazási területe az interpoláció [15], hiszen ha adott két egymást követő kép egy sorozatból, akkor a becsült optikai áramlással köztes fázisok generálhatóak az alábbi képletek valamelyikével:

$$I_{interpolated}(x, y, t + dt) = I_2(x + v \cdot dt, y + u \cdot dt, t + dt) \tag{2.11}$$

$$I_{interpolated}(x, y, t + dt) = I_1(x - v \cdot dt, y - u \cdot dt, t + dt) \tag{2.12}$$

Ezt az eljárást alkalmazták például orvosi képfeldolgozás területén egyes szervek, tumorok mozgásának becslésére [16] és modellezésére, időbeli [17] vagy akár térbeli [18] felbontás növelésére , de akár videók esetében is jól alkalmazható, köztes frame-ek generálására [19], ezáltal növelve a képfrekvenciát és javítva/folytonosabbá téve a mozgást.

Az időbeli interpolációval analóg módon az eljárás alkalmazható például térben is, ahol az időkoordináta szerepét a képsíkra merőleges z koordináta veszi át. Ezáltal a szomszédos szeletekre lehet meghatározni az optikai áramlást, ennek segítségével pedig köztes szeletek közelíthetőek.

A módszer nagy előnye lehet, hogy az egyszerű lineáris interpolációval ellenben azokban az esetekben is pontosabban tudna működni, ahol az egyes szeletek között egy konstans eltolás van, illetve a nem átlapolódó kontúrok esetét is az intuíciónak

megfelelően kezelné. Egyszerre lenne képes az eltolást és a kontúr megváltozását is figyelembe venni, hiszen az áramlást pixelenként becsli.

A hátránya viszont, hogy a becslés pontosságához jól textúrázott környezet szükséges, amely nem feltétlenül teljesül (célszerű a nyers szeletfelvételeket használni, ahol erre nagyobb esély van), valamint a korábbiaknak megfelelően kis elmozdulások esetén lesz igazán pontos és hatékony. Nagy elmozdulások esetén az említett problémák mellett a mintavételezés nehézsége is felmerül, abban az esetben, ha a kép értelmezési tartományán kívülről kell a képet mintavételezni (távolságtranszformáció esetén ez még kezelhető lenne, bár költségesen). A szimmetria sem biztosított két szelet között számított áramlásra, ahhoz további megkötések figyelembe vétele szükséges.

2.5. Radon-transzformáció alapú megközelítés

Egy másik lehetséges megközelítése a feladatnak a Radon-transzformáltak [20] eloszlásainak interpolálásán keresztül történik, a [21] cikkben leírtak szerint. A módszer motivációja egy folytonos és sima átmenet képzése akár 2D, akár 3D eloszlások között, amely egyenletesen viszi át az egyiket a másikba.

Az eljárás lényege, hogy 2D eloszlások esetén képezi azok Radon-transzformáltját, amely nem más, mint az eloszlás különböző irányokból vett vetületeinek összessége az alábbi képlet szerint:

$$p_\theta(t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) \cdot \delta(x \cdot \cos(\theta) + y \cdot \sin(\theta) - t) dx dy \tag{2.13}$$

Ahol $f(x,y)$ a 2D eloszlás, θ annak az egyenesnek az x tengellyel bezárt irányszöge, amelyre a vetületet számítjuk.

Ezek mindegyike egy-egy 1D eloszlás, amelyek közötti interpoláció a *distribution interpolation* [22] technikával lehetséges egy normalizálás után. Az interpoláció eredményeként, köztes 1D eloszlások keletkeznek, amelyeket a normalizálás miatt vissza kell skálázni annak megfelelően, hogy az összintegrálok folytonosan menjenek át egyikből a másikba. Ezeken, mint egy köztes fázis Radon-transzformáltján, vég-

rehajtható a transzformáció inverze, a szűrt visszavetítés (*filtered back projection*) [23] és így generálható egy köztes 2D eloszlás is. A módszer jól általánosítható 3D eloszlások esetére is, amikor annak 2D vetületeit kell képezni és azok között a már ismertetett módszerrel interpolálni.

Az algoritmus jól párhuzamosítható és egyértelműen jól tudja kezelni azokat az eseteket, amelyeket az alaktartó interpoláció nem. Mindehhez nem kell a távolság-transzformáltakat sem képezni, egyszerűen a nyers adatokon operál, illetve nem kell a problémát nagyobb dimenzióba beágyazni a megoldáshoz, mint [2] cikkben leírt esetben.

2.6. Jellemző alapú megközelítés

A jellemző alapú megközelítés nem vált reprezentációt a két kontúr között interpoláló felület generálásához, hanem közvetlenül a kontúrok direkt geometriai modelljeivel operál. Ahogyan a korábbiakban már utaltam rá, pontonkénti megfeleltetés szükséges ahhoz, hogy egy háromszöghálót, mint 3D modellt lehessen generálni és megjeleníteni [9], [24]. A módszerrel nehezebbé válik a különböző topológiájú görbék közötti interpoláció, hiszen ebben az esetben nem minden görbének akad párja a megfeleltetés szempontjából, ami a köztes fázisok generálásához, a modell készítéséhez viszont szükséges.

A módszer kombinálható akár az implicit reprezentáció alkalmazásával is. Ebben az esetben is szükséges a görbe pontjai közötti pontonkénti megfeleltetés, amely történhet például az ismert DTW [25] vagy CDTW [26], [27] algoritmusok valamelyikével. Ezzel analóg módon definiálható a detektált kontúrok közötti megfeleltetés is, amely a különböző görbék összerendelését végzi el egy bizonyos szabályrendszer alapján (például kontúrok távolságai, hasonlóságuk mértéke stb.). Abban az esetben, ha ez adott, akkor a köztes adatok egyszerűen generálhatóak, ha az egyes görbék pontjait folytonosan visszük át egyikből a másikba a megfeleltetések eredménye alapján, és erre az elrendezésre számítjuk ki a távolság-transzformáció értékét.

3. fejezet

A számítógépes feldolgozás lépései

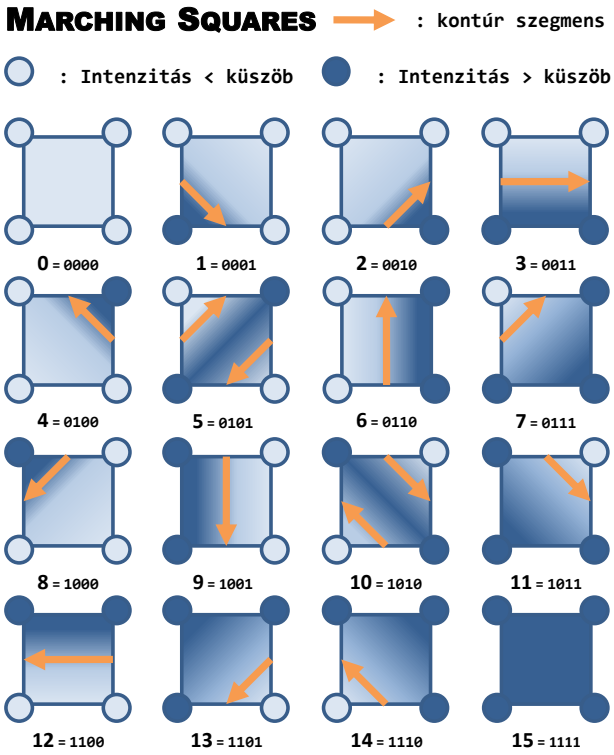
3.1. A kontúrok detektálása

A kontúrdetektáláshoz használt *Marching Squares* ([1]) algoritmus bementét egycsatornás, szürkeárnyalatos szeletfelvételek képezik. Ezen felül az algoritmusnak egyetlen további bemeneti paramétere van, amely a küszöbértéket definiálja.

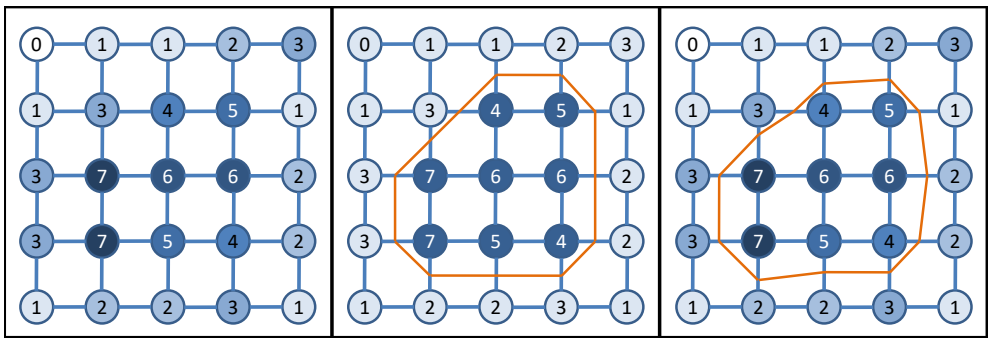
Az algoritmus a képet oly módon pásztázza végig, hogy mindig négy szomszédos (2×2) pixel értékét olvassa egyszerre, a 3.1 ábrának megfelelően, majd azt vizsgálja, hogy egyes intenzitás értékek a megadott küszöbérték alatt vagy felett helyezkednek el. Az osztályozás eredményeként generálható egy kód mind a 16 lehetséges esethez úgy, hogy a vizsgált négyzet sarkait kettes számrendszerben négy különböző helyiértéknek feleltetjük meg.

A 16 különböző esetből topológiailag csak négy különbözik egymástól, melyeket egy look-up táblázatban tárolhatunk. A módszer lényege, hogy a képeken úgy keresünk kontúrokat, hogy az egyes 2×2-es cellákat véve, minden olyan élt vizsgálunk, amelyhez tartozó egyik csúcspontra igaz, hogy az ott található intenzitásérték a küszöb alatti, míg a másik a küszöb feletti. Lineáris interpoláció segítségével megkeressük az adott él mentén azt a pontot, ahol az intenzitás értéke éppen a küszöbértékkel egyezik meg. A metszéspontok kiszámításával beszúrható egy-egy kontúrszegmens, ami a metszéspontokat köti össze. Ezt a folyamatot mutatja be a 3.2 ábra.

Érdekes és nem egyértelmű eseteket képeznek az ötös és tízes lehetőségek. Ilyenkor nem egyértelmű, hogy a két-két szegmens beszúrása milyen módon kell, hogy történjen, de ennek feloldására is van lehetőség. Az egyik, hogy vizsgáljuk a cella



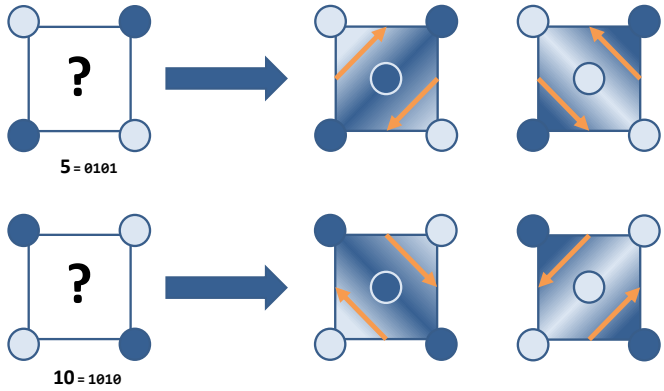
3.1. ábra. A Marching Squares algoritmus bemenetén elforduló 16 lehetséges eset. Az ábra a kontúrszegmensek irányítását is szemlélteti.



3.2. ábra. A Marching Squares algoritmus lépéseinek szemléltetése. A bemenet egy intenzitáskép, aminek minden 2×2-es cellájához meghatározható a 16 lehetséges eset valamelyike. Csúcspontokat azokban a pozíciókban szúrunk be, ahol a lineáris interpoláció a küszöbértékkel azonos eredményt ad.

közepére számolt intenzitásértéket (a cella sarkaiban vett értékek alapján ez a négy intenzitás átlagának fog megfelelni) és az alapján, hogy ez a küszöbérték alatt vagy felett található, oldható fel a probléma és határozható meg a konnektivitás a 3.3 áb-

rának megfelelően. Habár a konkrét gyakorlati alkalmazásban nem valószínű, hogy e két eset valamelyike is előfordulna, kezelésükre érdemes felkészülni.

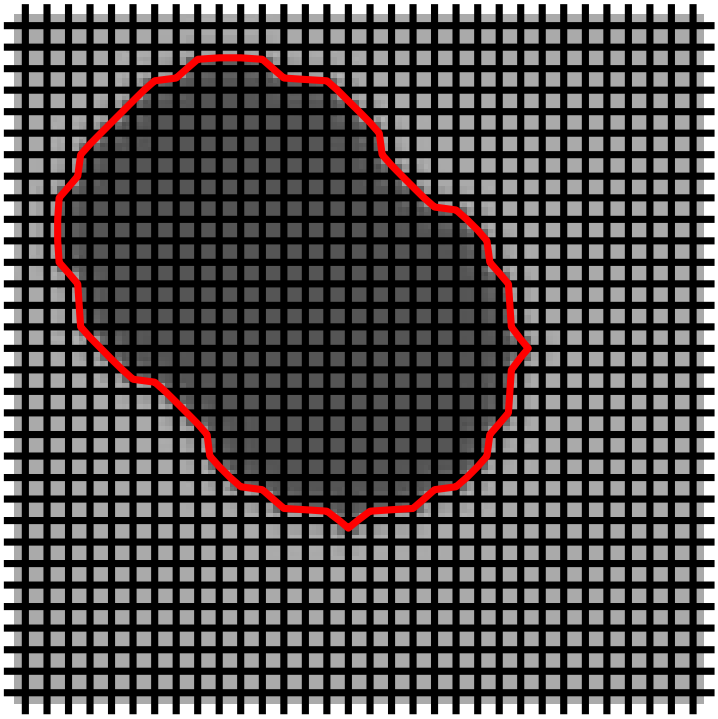


3.3. ábra. A Marching Squares algoritmus ötös és tízes, nem egyértelmű eseteinek feloldása a középpontra számított intenzitás segítségével.

A Marching Squares minden esetben zárt görbéket talál a képen (akár többet is), ha tudjuk azt garantálni, hogy a kép széleinél minden intenzitásérték a küszöbértéknél nagyobb/kisebb legyen az objektum környezethez viszonyított intenzitásától függően. Ez könnyen megtehető például a képek peremezésével.

A fent ismertetett algoritmus nagy előnye, hogy jól párhuzamosítható és ezáltal a végrehajtás felgyorsítható, hiszen az egyes cellákra végzett műveletek egymástól függetlenek és nem tartalmaznak elágazást. Eredményként önálló kontúrszegmensek sorozatát kapjuk, amelyeket két végpontjuk definiál. Az élek ilyen módon történő tárolása redundáns, hiszen minden pont kétszeresen lesz eltárolva, egyszer, mint kezdő- és egyszer, mint végpont. Ez a probléma egy utófeldolgozás során megoldható és a pontok kigyűjthetőek egy olyan listába, amely azokat egy meghatározott sorrendben tárolja a konnektivitásnak megfelelően, ezáltal egy egybefüggő görbét kapunk.

Mivel a következő fázisban előjeles távolság-transzformációt szeretnénk számítani, ezért fontos még az is, hogy az egyes pontokat milyen sorrendben illesztjük be a listába, azaz az egyes szegmenseknek milyen irányítást adunk. Ha ügyelünk arra, hogy a küszöbérték feletti értékű pixelek mindig a kontúr egy adott oldalára essenek, akkor egy irányított görbét kapunk eredményül, amely esetén jól meghatározható, hogy egy adott pont a görbe belsejében, vagy azon kívül található. Konkrét esetre mutat példát a 3.4 ábra.



3.4. ábra. A kontúr detektálása egy konkrét esetben Marching Squares algoritmussal.

3.2. A távolság-transzformáció kiszámítása

Az előjeles távolság-transzformáció (SDT: *Signed Distance Transform*) számítása a következő lépés. A feladat, hogy egy adott felbontású rácsháló (kép) esetében az egyes rácspontoknak (pixeleknek) a kontúrtól vett előjeles távolságát határozzuk meg. Bináris képek esetén sokszor alkalmazzák ezt a transzformációt, amire hatékony algoritmusok is születtek már. Példa erre Felzenszwalb és Huttenlocher lineáris futási idejű algoritmusa [28]. Jelen esetben, mivel nem egy bináris képből indulunk ki, hanem a Marching Squares algoritmussal detektált kontúrokból, így pontosabban is meghatározható az SDT, ha minden pontban kiszámítjuk az egyes szegmensektől, illetve azok végpontjaitól számított távolságok minimumát.

Felmerülő kérdés azonban, hogy mi a teendő abban az esetben, ha a SDT számításhoz használt kép felbontása nem egyezik meg a bementi kép felbontásával, amin a kontúrokat detektáltuk. Ez az eset azért fontos, mivel általában a bemeneti képek nagyobb felbontásúak, viszont nem szükséges az SDT-t ugyanekkora felbon-

tásban megkapni, elegendő kisebb felbontással kiértékelni, ami jelentősen csökkenti a számítási költséget.

A Marching Squares algoritmus pixelkoordinátákban határozza meg a kontúr egyes pontjait. Az SDT számításánál viszont tetszőleges felbontású képre (akár kisebbre is) végezzük el a számításokat. Legelőször ennek a transzformációhoz tartozó képnek a rácspontjait kell a [0, 1] intervallumon keresztül áttranszformálni az eredeti kép koordinátarendszerébe, hiszen a kontúrpontokat itt számítottuk. Ez egyszerűen megtehető a kiindulási kép felbontásának ismeretében.

$$x_{norm} = \frac{x_{px,1}}{width_{px,1}} \tag{3.1}$$

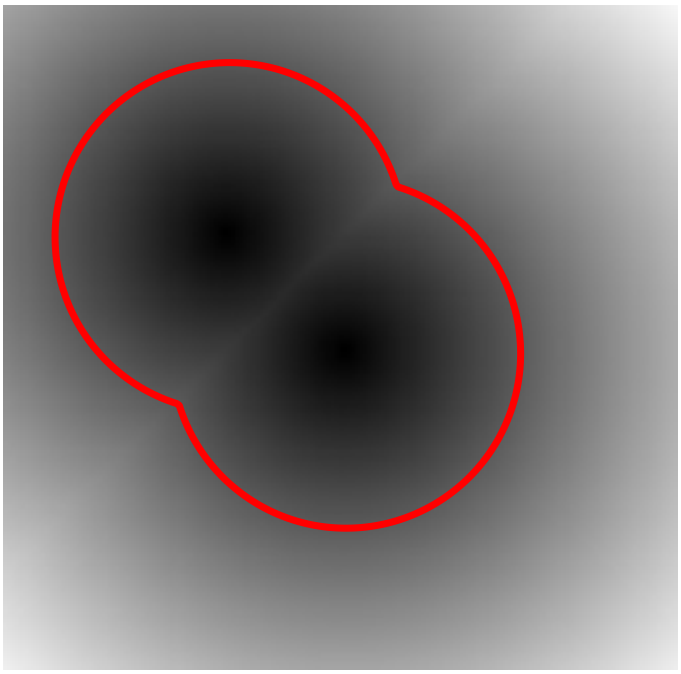
$$y_{norm} = \frac{y_{px,1}}{height_{px,1}} \tag{3.2}$$

$$x_{px,2} = x_{norm} \cdot width_{px,2} \tag{3.3}$$

$$y_{px,2} = y_{norm} \cdot height_{px,2} \tag{3.4}$$

Egy egyszerű és *brute force* implementáció lehet, hogy a célkép egyes pixeleit véve, számítjuk azok távolságát az egyes szakaszoktól, illetve a végpontoktól, és keressük a legkisebb abszolút értékű távolságot. Az előjelet pedig egyszerűen skaláris szorzással határozhatjuk meg, mivel az előző lépésben a görbét irányítással együtt definiáltuk. Az algoritmus lépésszáma nem túlságosan kedvező, hiszen $O(N \times m)$-es, ahol N a rácsháló pontjainak számát, m pedig a szakaszok számát jelöli.

Ennél hatékonyabb implementáció is kidolgozható, illetve a meglévő is párhuzamosítható, hiszen itt sincs összefüggés az egyes rácspontokban végzett számítások között. Egyik lehetőség például a kontúrpontok számának csökkentése olyan szakaszokon, ahol görbe simább, görbülete kisebb, tehát nincsenek nagyfrekvenciás komponensek. Egy másik lehetőség például a KD-fa [29] alkalmazása, amelyben hatékonyan és gyorsan található meg a legközelebbi kontúrpont, ezzel pedig szűkíthető a keresési tartomány a távolság-transzformáció meghatározásakor.



3.5. ábra. Előjeles távolság-transzformáció kiszámítása egy konkrét példa esetén.

4. fejezet

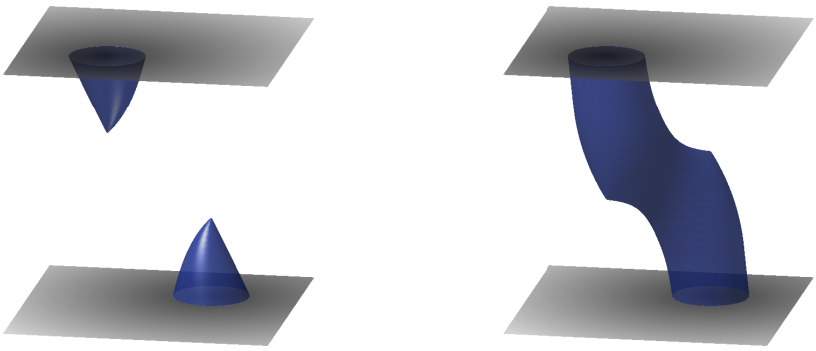
Nem átlapolódó kontúrok közötti intepoláció

Nem átlapolódó kontúrok esetén a távolság-transzformációk közötti egyszerű lineáris interpoláció nem eredményez folytonos átmenetet, hiszen az egyes szeleteken számított pozitív és negatív távolságértékek között mindenképpen nulla értéket kapunk valahol a két szelet közötti régióban. Ez az egyes nem átlapolódó felületelemek lezárását eredményezi, amely gondot jelenthet például nagy görbületű felületek modellezése esetén, ahol nem ez az elvárt eredmény. Ez látható a 4.1. ábrán is, összehasonlítva a dolgozatban kifejtett új módszerrel kapott megoldással.

4.1. Az új módszerek leírása

4.1.1. 2D eset

A probléma megoldására egyszerű és számítási költségeket tekintve olcsó megoldást jelenthet az alábbi eljárás. Ha ismert előre az a maximális távolság, melyre egymáshoz képest ennél kisebb távolságra elhelyezkedő felületelemek között szeretnénk folytonos átmenetet képezni, akkor először az összekötni kívánt kontúrokat kell kiterjeszteni. Mivel az implicit reprezentációban számunkra a zérus szintfelület definiálja a modellezni kívánt felületet, ezért legegyszerűbben a távolságértékek eltolásával tehetjük meg a kontúrok kiterjesztését. Amennyiben a távolság-transzformáció a görbe belsejében vesz fel negatív értékeket, úgy az eltolás negatív irányba, ellenkező



4.1. ábra. Szomszédos szeleteken nem átlapolódó keresztmetszeti kontúrok közötti interpoláció különböző módszerekkel. Balra: egyszerű lineáris interpoláció a távolság transzformáltak között, jobbra: saját módszer alkalmazása.

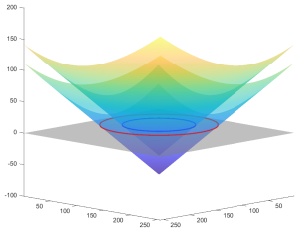
esetben pozitív irányba kell, hogy történjen. A kiterjesztésre olyan mértékben van szükség, hogy az összekötni kívánt kontúrok között mindenképpen legyen átlapolódás az eltolás után. Ezt szemléltetik a 4.2a és a 4.2b ábrák valamint ezt írják le az alábbi képletek. f_1 és f_2 a két ismert távolságmező, D az eltolás mértéke, C a kontúrok halmaza, $d(.,.)$ pedig a geometriai távolság két kontúr között, $OF > 0$ az átlapolódás mértékét meghatátozó faktor (a bemutatott példákban értéke 0.25).

$$f_1'(\mathbf{x}) = f_1(\mathbf{x}) - D \tag{4.1}$$

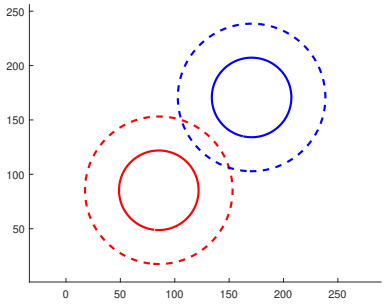
$$f_2'(\mathbf{x}) = f_2(\mathbf{x}) - D \tag{4.2}$$

$$D = \max_{c_1,c_2 \in C} \{d(c_1, c_2)\} \cdot 0.5 \cdot (1 + OF) \tag{4.3}$$

Ha a korábbiakban elmondottakhoz hasonlóan feltételezzük a legnagyobb olyan távolságot, amelyek között még folytonos átmenetet szeretnénk képezni, úgy e távolságérték felénél nagyobb mértékben kell mindkét implicit függvényt módosítani, annak érdekében, hogy az eljárás teljesen szimmetrikus legyen. Ezek után a távolságtranszformációk között már egyszerűen lineárisan interpolálva is folytonos átmene-



(a) A távolságmező eltolásának hatása a zérus szintgörbére, amely nagyításnak, vagy zsugorításnak feleltethető meg.



(b) A nem átlapolódó kontúrok nagyításának szemléltetése.

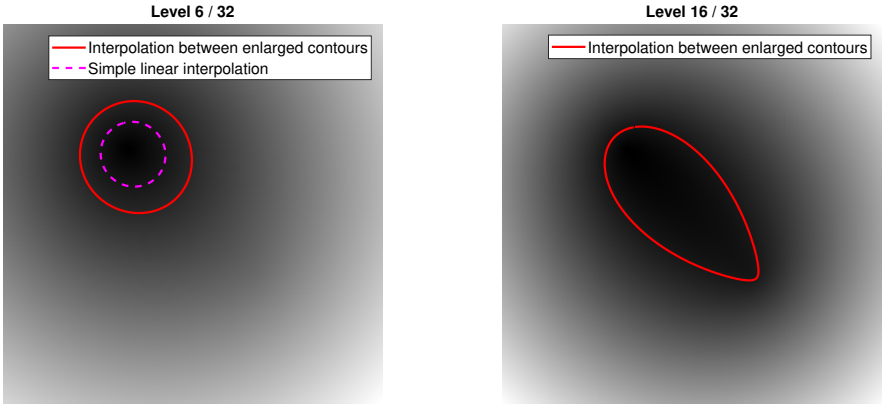
4.2. ábra. Az egyes kontúrok kiterjesztésének művelete.

tet kapunk, hiszen a problémát visszavezettük arra az esetre, mikor az átlapolódás biztosított. Ezt az esetet az implicit reprezentációnak köszönhetően már rendkívül egyszerűen kezelhetjük, pontonkénti megfeleltetés nélkül is, és alkalmazásának ez az egyik nagy előnye.

Az eredmény ekkor azonban még nem lesz megfelelő, hiszen ha egységesen toljuk el a távolságmezőben szereplő értékeket és így interpolálunk, úgy az eredeti adatot is torzítjuk, amely nem célja az eljárásnak. Éppen ezért egy korrekció elvégzése szükséges minden interpolált szelet esetén. A megkötés, hogy a kezdeti- és végfázisokban az eredeti adatot kapjuk vissza, a torzításmentes kontúrokkal, míg a köztes szeletek esetén az eredeti mintavételi síkoktól távolodva egyre jobban kívánjuk érvényesíteni a torzítás hatását, megteremtve ezzel az átmenetet. Az eredeti szeletek esetén tehát a levont távolságértékkel kell korrigálni a távolságmezőt, így az eredeti implicit reprezentációt kapjuk vissza. A közbülső szeletek esetén pedig, ettől egyre kisebb mértékben kell a korrekciót elvégezni, egészen a középső szeletig, ahol a torzítás (nagyítás) hatása a leginkább érvényesül. Mindez egy korrekciós szorzófaktoron keresztül érvényesíthető. A lineáris interpolációt és a korrekciós faktorral (DCF - Distance Correction Factor) való korrigálást az alábbi képletek mutatják. Egy összehasonlítás a lineáris interpolációval két szelet esetére pedig a 4.3. ábrán látható.

$$f_i(\mathbf{x}) = \alpha \cdot f'_1(\mathbf{x}) + (1-\alpha) \cdot f'_2(\mathbf{x}) + DCF \cdot D \tag{4.4}$$

$$DCF = g(\alpha), \qquad \alpha \in [0,1] \tag{4.5}$$



4.3. ábra. A szomszédos távolságtérképek közötti egyszerű lineáris interpoláció eredményeként kapott közelítő távolságmezőn detektált kontúrok, valamint a saját megoldással generált köztes távolságmezőn detektált kontúrok.

Ezek után a feladat ennek a szorzófaktornak, mint a mintavételi síkoktól mért minimális távolság függvényének meghatározása, amely az $\alpha \in [0,1]$ intervallumon van értelmezve, $\alpha = 0$-ban és $\alpha = 1$-ben egységnyi értéket vesz fel, $\alpha = 0.5$-ben pedig zérust. Számos ilyen függvény definiálható, érdemes ezek hatásait a végeredményre átgondolni és megvizsgálni. Néhány lehetséges, a fentieknek megfelelő függvényt alább láthatunk, amelyek esetén az eredmények rendre a 4.4a és a 4.4b ábrákon figyelhetőek meg.

$$g_1(\alpha) = 2 \cdot |\alpha - 0.5| \tag{4.6}$$

$$g_2(\alpha) = 0.5 \cdot (\cos(2 \cdot \pi \cdot \alpha) + 1) \tag{4.7}$$

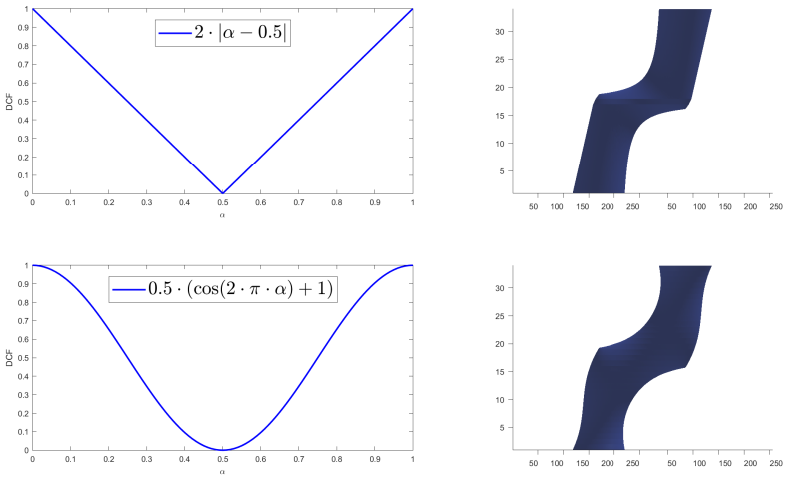
$$g_3(\alpha) = \max(\frac{|\alpha - 0.5| - \beta}{0.5 - \beta}, 0) \tag{4.8}$$

$$g_4(\alpha) = smoothL1(\alpha) = \begin{cases} \frac{1}{1-\beta} \cdot \frac{(\alpha-0.5)^2}{\beta} & |\alpha - 0.5| \leq \beta \\ \frac{|\alpha-0.5|-0.5\cdot\beta}{0.5\cdot(1-\beta)} & |\alpha - 0.5| \leq \beta \end{cases} \tag{4.9}$$

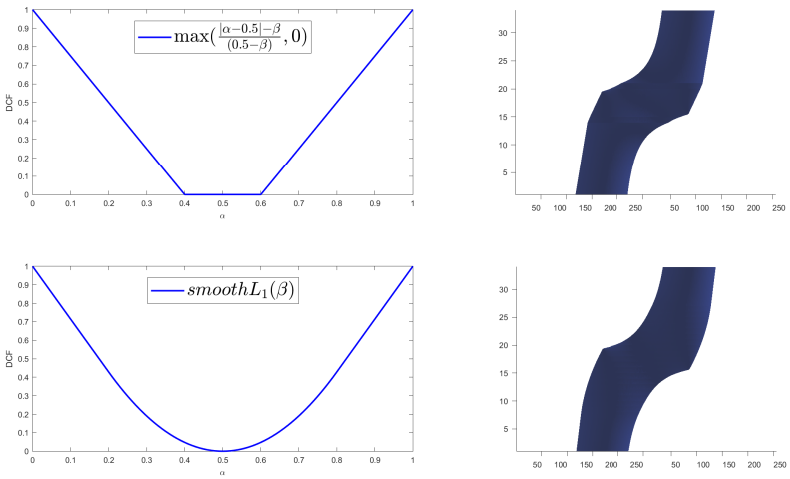
A 4.4a és a 4.4b ábákon megtekinthető eredményekből észrevehető, hogy célszerű a függvényt úgy megválasztani, hogy deriváltja az értelmezési tartomány szélein egységnyi legyen, hiszen ekkor az eredeti mintavételi síkok közelében a kontúr szinte alig változik az átmenet során. Ennek köszönhetően képezhető sima átmenet több mintavételi sík esetén a szomszédos tartományok között, ellenben a koszinuszos függvénnyel, amely két szomszédos tartomány határán törést eredményezne az átmenetben, hiszen ahogyan az eredményből is megfigyelhető a kezdeti szakaszon elvékonyodik a keresztmetszet.

Vizuálisan a legszebb eredményeket a $smoothL1$ függvény adta, amely az értelmezési tartományok szélein abszolútérték függvényként viselkedik, a közepén pedig négyzetes függvényként. A generált átmenet természetesen nem tökéletes, hiszen éles sarkok figyelhetőek meg rajta a tartomány közepén, azonban mégis képes volt a folytonos átmenetet megteremteni, amelyre az egyszerű interpoláció nem volt képes. A módszernek további hátránya, hogy átlapolódó kontúrok esetén, az átmenet megvastagodik. Ez annak köszönhető, hogy az algoritmus a görbék implicit reprezentációival operál, a geometriai reprezentációt nem veszi figyelembe, és ezáltal nem tesz különbséget, átlapolódó és nem átlapolódó kontúrok között. Mindezért cserébe, viszont a számítások egyszerűek, jól párhuzamosíthatóak és gyorsan elvégezhetőek.

A fent említett hiányosságra egyik kézenfekvő ötlet lehet, ha a korrekciót nem egyszerűen egy szorzófaktor segítségével végezzük el. A két kiindulási szeletet, mint kezdeti és végfázist tekintve, meghatározható a detektált kontúrok által közrezárt területek nagysága. A köztes szeleteken végzett korrekció számítható az alapján is, hogy megköveteljük a terület nagyságának lineáris átmenetét a kezdő állapotból a



(a) Abszolútérték és koszinuszos függvény



(b) Vágott abszolútérték és smooth L1 függvény

4.4. ábra. Különböző függvénykapcsolatok a korrekciós faktor (DCF) és az interpoláció α paramétere között, valamint az alkalmazásaik eredménye.

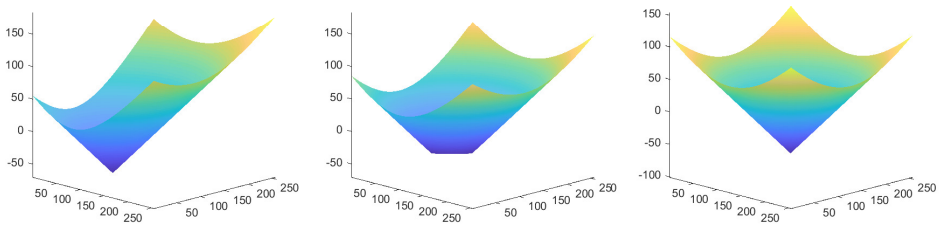
végállapotba. Ez szemléletesen úgy képzelhető el, hogy homogén eloszlást feltételezve, adott agyagmennyiséget a kezdeti fázisból a végfázisba egyenletesen változtatva viszünk át. A megoldás valóban jól kezeli azokat az egyszerű eseteket, mikor két átlapolódó kontúr között szeretnénk interpolálni, hiszen az összeköttetést se nem vastagítja meg, se nem vékonyítja el olyan módon, mint ahogyan az nemkívánatos lenne. Gondot jelentenek azonban már a topológiailag különböző esetek is akkor, ha mondjuk nincs átfedés két objektum között, valamint a kezdő és végfázisban is a területek megegyeznek egymással. Ekkor tulajdonképpen a korrekció ellehetetleníti a megfelelő működést.

Robusztusabb lehet az a megoldás, ha nem egyszerűen csak az implicit reprezentációkkal operálunk, hanem a korrekció után kapott interpolált távolságmezőn egy kontúrdeketálást hajtunk végre, majd újra számoljuk az ehhez tartozó távolságmezőt. Ennek azért lehet jelentősége, mert két távolság-transzformált közötti egyszerű interpoláció nem őrzi meg annak tulajdonságait, így nem lesz valóban távolságmező. A gradiensnek ugyanis minden olyan pontban, ahol a deriválás elvégezhető, egységnyinek kell lennie, az interpoláció azonban behozhat olyan régiókat, amelyekre nem teljesül ez a feltétel, és kialakulhatnak „platók". Ezek érzékenyebbé tehetik a rendszert abban az esetben, ha a platók mentén az interpolált érték éppen zérus. A fentieket szemlélteti a 4.5. ábra. Különböző esetekre vett megoldásokat mutat a 4.6., a 4.7. és a 4.8. ábra.

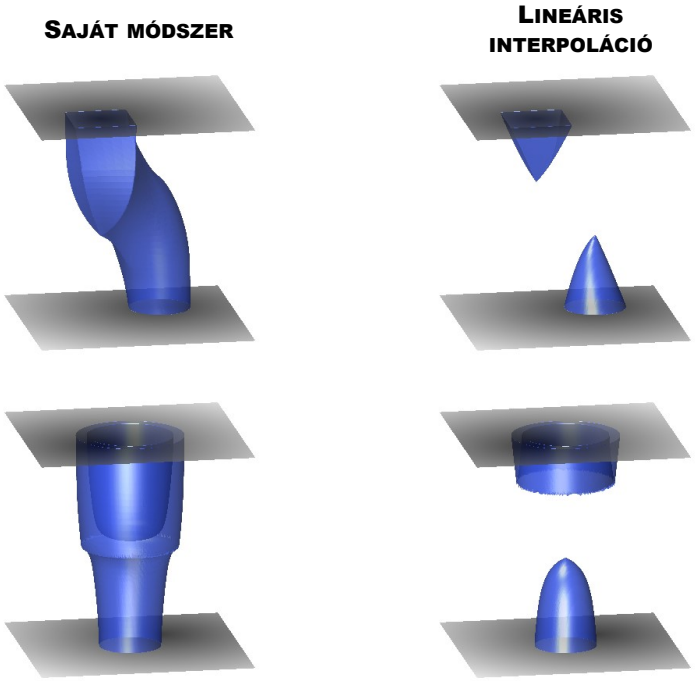
Az első javasolt módszer összefoglalása és lépései a következők:

1. Kontúrok detektálása az eredeti szeleteken
2. SDT kiszámítása az eredeti szeletekre
3. Eredeti távolságmezők eltolása = kontúrok kiterjesztése és átlapolódás garantálása
4. Lineáris interpoláció minden köztes szeletre
5. Egyes köztes szeleteken korrekció elvégzése az adott, szomszédos szeletektől mért minimális távolság függvényében

Számítási költségeket tekintve drágább megoldást jelenthet a következő megközelítés. Ha a korábbiakhoz hasonlóan vesszük az egyes szeleteket és a távolság-

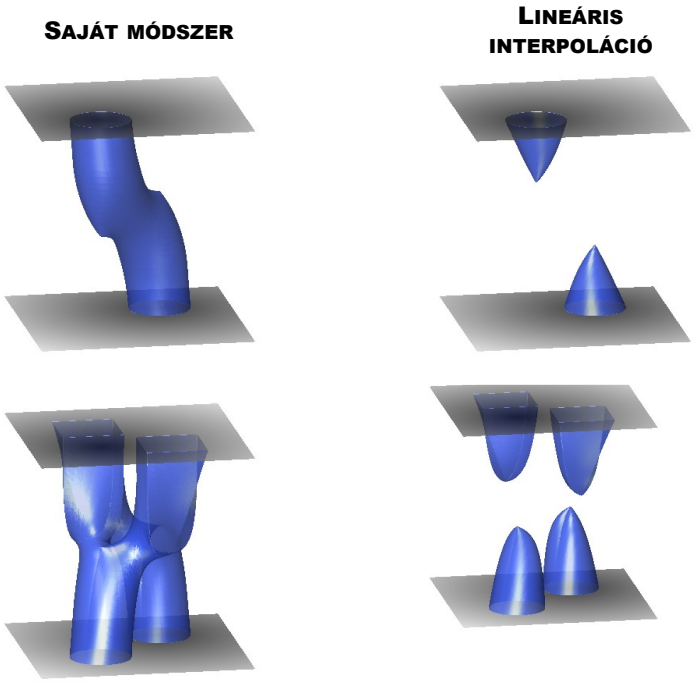


4.5. ábra. A távolságmezők, mint függvények ábrázolása 3D-ben a kezdeti- és végfázisokban (széleken), valamint a lineáris interpoláció segítségével kapott köztes közelítés (középen).



4.6. ábra. A dolgozatban bemutatott módszer alkalmazása különböző esetekre.

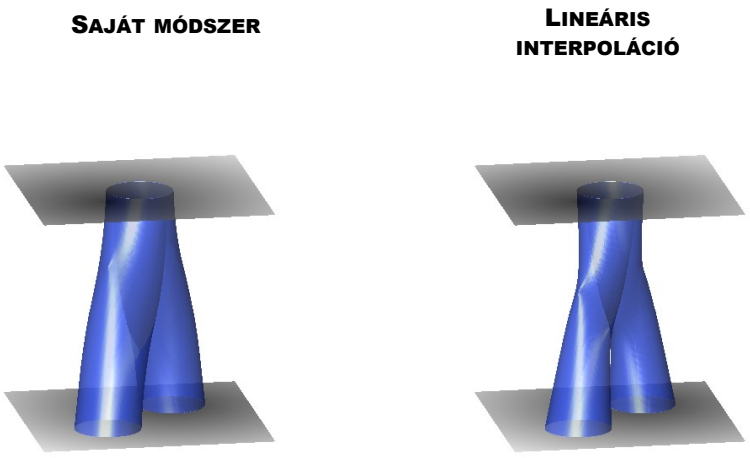
transzformáltjaik értékeit eltoljuk D-vel (ezzel kiterjesztve a kontúrokat), akkor ezek között interpolálva kapjuk az elsődleges közelítéseket. Az első javasolt módszer ese-



4.7. ábra. A dolgozatban bemutatott módszer alkalmazása különböző esetekre.

tében egy korrekciós szorzófaktoron keresztül végeztük el a távolság-transzformált visszatolását és így kaptunk folytonos összeköttetéseket. Ha ehelyett minden köztes közelítésre, amely nem rendelkezik a távolság-transzformáció tulajdonságaival (lásd „platók"), egy kontúrdetektálást végzünk el, és erre számítjuk ki az SDT-t, akkor már egy valódi távolságmezőt kapunk. Ha kontúrokat a korábbiakhoz képest nagyobb mértékben terjesztjük ki (például 25% helyett 75%-kal), akkor a korrekció közvetlenül a szorzófaktor nélkül is (értékét tehát konstans 1-re választva) elvégezhető, és így is folytonos átmenetet kapunk. Ezt mutatja a 4.9. ábra.

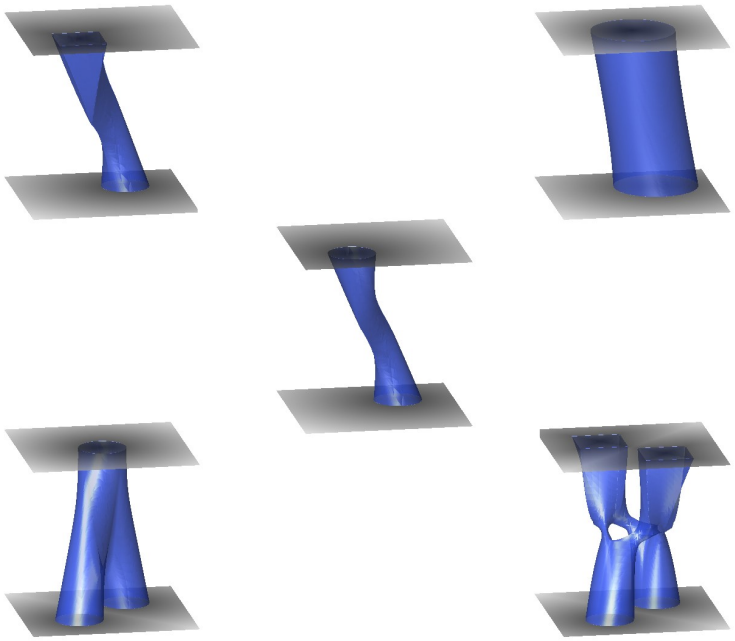
Ennek előnye, hogy simább az átmenet, nem tartalmaz nemkívánatos sarkokat és nyúlványokat, valamint nem szélesíti ki az átlapolódó kontúrok közötti átmenetet. Hátránya a nagy számításigény, hiszen minden köztes szeletre a kontúrok detektálása és egy-egy SDT kiértékelése szükséges. Nehezebben kezeli továbbá a topológiailag különböző eseteket is.



4.8. ábra. A dolgozatban bemutatott módszer alkalmazása különböző esetekre.

A második javasolt módszer összefoglalása és lépései a következők:

1. Kontúrok detektálása az eredeti szeleteken
2. SDT kiszámítása az eredeti szeletekre
3. Eredeti távolságmezők eltolása = kontúrok kiterjesztése és nagyobb átlapolódás garantálása
4. Lineáris interpoláció minden köztes szeletre
5. Minden köztes szeletre egy kontúrdetektálás elvégzése
6. Minden köztes szeletre egy SDT meghatározása a korábbi kontúrdetekció alapján
7. Minden köztes szeletre a korrekció elvégzése közvetlenül az eltolás értékének megfelelően (D-vel)



4.9. ábra. A köztes szeletek közelítése kontúrdetektálás és SDT számítás segítségével, miközben a kontúrokat nagyobb mértékben kell kiterjeszteni.

4.1.2. 3D eset

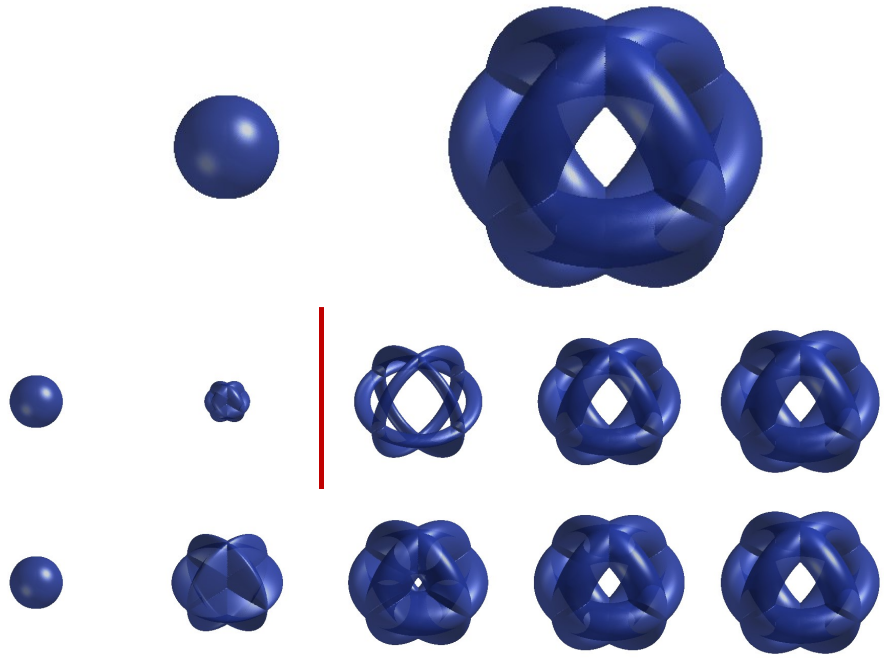
Mindkét eljárás 3D-be is kiterjeszthető metamorfózisok vizualizációjára. Adott két 3D távolság-transzformáció, amelyek között egy folytonos áttűnést szeretnénk generálni. Abban az esetben, ha ezek között van átlapolódás, úgy a probléma egyszerűen kezelhető több módszerrel is, mint például, az egyszerű lineáris interpolációval. Abban az esetben azonban mikor nincs átlapolódás, a legtöbb módszer használhatatlanná válik, vagy jelentős módosításra szorul. Az elsőnek bemutatott eljárással azonban hasonlóan szép eredményekhez juthatunk, mint a 2D esetben.

Például egy hármas tórusz és a belsejében átlapolódás mentesen elhelyezett gömb közötti áttűnés lineáris interpolációval az alábbiak szerint alakulna: a gömb először fokozatosan eltűnne, majd hirtelen megjelenne a hármas tórusz, amely azt követően növekedne a végleges méretére.

Az átlapolódás mesterséges garantálásával azonban már egy fizikailag is plauzibilisebb (ám fizikai tartalommal/modellező képességgel nem bíró) áttűnés ge-

nerálható. Ehhez mindösszesen a két objektum közti minimális távolság ismerete szükséges.

A 3D esetben a két bemutatott módszer közül a korábbit alkalmaztam. Ennek oka a lényegesen egyszerűbb műveletek és az átmenet fizikailag plauzibilisebb lefolyása. Ezt mutatja a 4.10. ábra. Az objektumok "anyagáram" szerűen mennek át egymásba, míg a második, költségesebb módszer esetén ez nem így lenne. További érv az első módszer alkalmazása mellett, hogy 3D-ben egy *marching cubes* algoritmus és egy távolság-transzformáció kiértékelése sokkal költségesebb a nagyobb dimenzió miatt, mint az elemi műveletek. A metamorfózisokat szemléltető további videó animációk az alábbi Google Drive linken érhetőek el: `https://drive.google.com/drive/folders/1zpVcDRNUwKJ2grPCxMDMat6C1AoQzdXq?usp=sharing`.



4.10. ábra. Egy gömb és egy hármas tórusz közötti átmenetet szemléltető ábra. Első sor: kiindulási és végállapotok. Középső sor: átmenet egyszerű lineáris interpoláció segítségével. A piros szakasz jelzi, hogy az átmenet nem folytonos, és bizonyos α értékekre nincs objektum a képen. Alsó sor: átmenet a dolgozatban tárgyalt új módszerrel.

5. fejezet

Összefoglalás és jövőbeli célok

A dolgozatban két új algoritmust mutattam be, amelyek képesek kiküszöbölni az implicit reprezentáció alkalmazásának egyik gyakori hátrányát. Az egyik számítási igényeit tekintve egyszerűbb és jól párhuzamosítható, míg a másik költségesebb, ám bizonyos esetekben jobb megoldást szolgáltat. A módszerek képesek nem átlapolódó régiók között is folytonos átmenetet biztosítani, amelyre a korábbi megközelítések nem adtak lehetőséget.

Mindemellett az egyszerűbb eljárás nem csak 2D-ben ad megoldást a problémára, de 3D-ben is lehetővé teszi a különböző, nem átlapolódó régiók közötti időbeli áttűnéseket, metamorfózisokat, amelyek egyúttal látványosak és az intuíciónak is megfelelnek.

A jövőben cél lehet a módszer GPU-ra történő adaptációja, amely akár valós idejű megjelenítést is lehetővé téve segíthetné az eljárás távhatást szabályozó paraméterének hangolását.

Irodalomjegyzék

[1] Marching cubes and variants. http://web.cse.ohio-state.edu/~wenger.4/publications/isosurface_book_preview.pdf, 2020.

[2] Greg Turk and James F. O'Brien. Shape transformation using variational implicit functions. In *Proceedings of ACM SIGGRAPH 1999*, pages 335–342, August 1999. URL http://graphics.cs.berkeley.edu/papers/Turk-STU-1999-08/.

[3] Adrian Bors, Lefteris Kechagias, and Ioannis Pitas. Binary morphological shape-based interpolation applied to 3d tooth reconstruction. *IEEE Transactions on Medical Imaging*, 21:100–109, 03 2002. DOI: 10.1109/42.993129.

[4] Roberto de Alencar Lotufo and Alexandre Xavier Falcao. Shape-based interpolation methods applied to medical imaging. *Anais do SIBGRAPI VI*, pages 323–331, October 1993.

[5] J.P. Lewis, Frédéric Pighin, and Ken Anjyo. Scattered data interpolation for computer graphics. *ACM SIGGRAPH 2014 Courses, SIGGRAPH 2014*, 01 2010. DOI: 10.1145/1900520.1900522.

[6] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. ACM '68, page 517–524, New York, NY, USA, 1968. Association for Computing Machinery. ISBN 9781450374866. DOI: 10.1145/800186.810616. URL https://doi.org/10.1145/800186.810616.

[7] Rao Garimella and Rao Garimella. Title: A simple introduction to moving least squares and local regression estimation intended for: A simple introduction to moving least squares and local regression estimation, 06 2017.

[8] Vaclav Skala. A practical use of radial basis functions interpolation and approximation. 37:137–145, 01 2016.

[9] David Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *ACM Trans. Graph.*, 11(3):228–258, July 1992. ISSN 0730-0301. DOI: 10.1145/130881.131213. URL https://doi.org/10.1145/130881.131213.

[10] David J. Fleet and Y. Weiss. Optical flow estimation. In *Handbook of Mathematical Models in Computer Vision*, 2006.

[11] Berkay Kanberoglu, Dhritiman Das, Priya Nair, Pavan K. Turaga, and David H. Frakes. An optical flow-based approach for minimally-divergent velocimetry data interpolation. *CoRR*, abs/1812.08882, 2018. URL http://arxiv.org/abs/1812.08882.

[12] Bruce Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision (ijcai). volume 81, 04 1981.

[13] Berthold Horn and Brian Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 08 1981. DOI: 10.1016/0004-3702(81)90024-2.

[14] Jean yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.

[15] Walter Hinterberger and Otmar Scherzer. Models for image interpolation based on the optical flow. *Computing*, 66(3):231–247, 2001. DOI: 10.1007/s006070170023. URL https://doi.org/10.1007/s006070170023.

[16] Heinz Handels, René Werner, Thorsten Frenzel, Dennis Säring, Wei Lu, Daniel Low, and Jan Ehrhardt. *Improved Reconstruction of 4D MSCT Image Data and Motion Analysis of Lung Tumors Using Non-linear Registration Methods*, volume 14, pages 2288–2291. 01 2007. ISBN 978-3-540-36839-7. DOI: 10.1007/978-3-540-36841-0_577.

[17] Jan Ehrhardt, Dennis Säring, and Heinz Handels. Interpolation of temporal image sequences by optical flow based registration. pages 256–260, 03 2006. ISBN 3-540-32136-5. DOI: 10.1007/3-540-32137-3_52.

[18] Ahmadreza Baghaie and Zeyun Yu. An optimization method for slice interpolation of medical images. *CoRR*, abs/1402.0936, 2014. URL http://arxiv.org/abs/1402.0936.

[19] S Lyasheva, R Rakhmankulov, and M Shleymovich. Frame interpolation in video stream using optical flow methods. 1488:012024, mar 2020. DOI: 10.1088/1742-6596/1488/1/012024. URL https://doi.org/10.1088/1742-6596/1488/1/012024.

[20] Johann Radon. On the determination of functions from their integral values along certain manifolds. *IEEE Transactions on Medical Imaging*, 5(4):170–176, 1986. DOI: 10.1109/TMI.1986.4307775.

[21] Márton József Tóth and Balázs Csébfalvi. Shape transformation of multidimensional density functions using distribution interpolation of the radon transforms. In *2014 International Conference on Computer Graphics Theory and Applications (GRAPP)*, pages 1–8, 2014.

[22] A.L Read. Linear interpolation of histograms. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 425(1):357–360, 1999. ISSN 0168-9002. DOI: https://doi.org/10.1016/S0168-9002(98)01347-3. URL https://www.sciencedirect.com/science/article/pii/S0168900298013473.

[23] Marwa Al and Mohammed Al-Hayani. The use of filtered back projection algorithm for reconstruction of tomographic image. pages 151–156, 01 2014.

[24] Y Dong and G.R Hillman. Three-dimensional reconstruction of irregular shapes based on a fitted mesh of contours. *Image and Vision Computing*, 19(3):165–176, 2001. ISSN 0262-8856. DOI: https://doi.org/10.1016/S0262-8856(00)00065-2. URL https://www.sciencedirect.com/science/article/pii/S0262885600000652.

[25] Zheng Zhang, Romain Tavenard, Adeline Bailly, Xiaotong Tang, Ping Tang, and Thomas Corpetti. Dynamic time warping under limited warping path length. *Information Sciences*, 393, 02 2017. DOI: 10.1016/j.ins.2017.02.018.

[26] Alon Efrat, Quanfu Fan, and Suresh Venkatasubramanian. Curve matching, time warping, and light fields: New algorithms for computing similarity between curves. *Journal of Mathematical Imaging and Vision*, 27:203–216, 04 2007. DOI: 10.1007/s10851-006-0647-0.

[27] Mario Munich and Pietro Perona. Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification. *Proceedings of the IEEE International Conference on Computer Vision*, 1, 02 1970. DOI: 10.1109/ICCV.1999.791205.

[28] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance transforms of sampled functions. *Theory of Computing*, 8(19):415–428, 2012. DOI: 10.4086/toc.2012.v008a019. URL http://www.theoryofcomputing.org/articles/v008a019.

[29] Kd trees. https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote16.html, 2021.