ACÉLVÁZAS SZERKEZETEK TERVEZÉSÉNEK AUTOMATIZÁLÁSA TOPOLÓGIAI OPTIMÁLÁSSAL

AUTOMATIZATION OF DESIGN OF FRAME AND TRUSS STRUCTURES USING TOPOLOGY OPTIMIZATION

Daróczy László, Dr. Jármai Károly***

ABSTRACT

Although the field of topology optimization is a well-researched and advanced area, using topology optimization is still only the starting phase for a design process as almost no manufacturing constraints can be taken into account. The current article proposes a method to fully automatize the design of truss and frame structures using topology optimization, image interpretation and finally through a sizing problem, and focuses especially on the image interpretation part by introducing a flexible and fast method to extract the cross-section and number of joints, and the number, location and type of connections between them. Finally, some examples are presented in 2D and 3D for the image interpretation.

1. BEVEZETÉS

A számítógépek fejlődésével, a rendelkezésre álló számítási kapacitásoknak és a növekvő teljesítményeknek hála, valamint a véges-elem modellek (a továbbiakban VEM) fejlődésének és elterjedésének hála a rácsos tartók tervezése topológiai optimálás segítségével (a továbbiakban *TO*) egy megvalósítható tervezési alternatívává vált. A manapság elterjedt topológiai optimáló szoftverek képesek egyszerre több peremfeltétellel és akár több célfüggvényre vonatkozó megszorítással dolgozni egyszerre [1,2]. Ennek ellenére a topológiai optimálás eredményei nem használhatóak fel rögtön közvetlenül, hanem csupán egy kezdőpontként fog szolgálni a tervezést végző mérnök számára, mivel a topológiai optimálás végeredménye pl. nem szabványos profilú, de sokszor akár nem is állandó keresztmetszetű gerendákból van összeállítva, illetve a kötések vagy kihajlásra típusára is nehézkes megkötéseket megadni. Bár több módszer is létezik optimális rácsos szerkezetek létrehozására, de a legtöbb ezek közül kötött számú rudat alkalmaz, amelyek csak előre meghatározott módon kapcsolódhatnak egymáshoz [3,4], és a CAD rendszerek és *TO*-rendszerek megfelelő integrációja még mindig hiányzik.

2. TERVEZÉSI FOLYAMAT AUTOMATIZÁLÁSA

Az új megközelítés alapötlete az, hogy a topológiai optimálás eredményét kizárólag az optimális topológia meghatározásához (azaz a szerkezetei elemek száma és típusa, valamint a közöttük lévő kötések) használjuk fel. A program automatikusan detektálja szerkezetet, és a végeredményt egy méretezési feladathoz használja fel bemenetként. A hasonló többlépcsős feldolgozások az az utóbbi időben egyre több helyen jelentek meg, ezek egy része azonban az alakoptimálás segítségével lép tovább [5,6,7], vagy egy szűk gyártási területre fókuszál, jellemzően kis méretű alkatrészekre szűkítve a kört [8]. Az alábbi megközelítés kifejezetten a nagyméretű acélvázas szerkezetekre fókuszál.

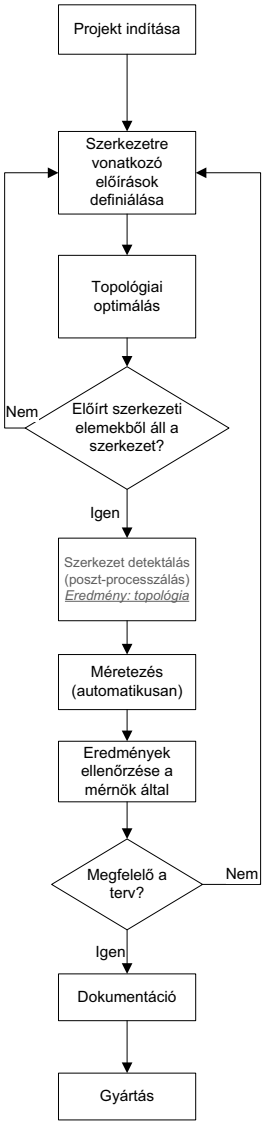
Azzal a feltételezéssel élve, hogy a topológia maga optimális, a méretezés célja pl. rácsos szerkezetek esetén a csomópontok pontos helyzetének és a rudak keresztmetszetének meghatározására egyszerűsödik. Ez utóbbi terület egy széles körben elterjedt eljárás a mérnöki tervezésben [9].

Mivel a méretezési feladat során a változók száma jelentősen kisebb, mint a topológiai optimálás folyamán, ezért ez gyorsan és pontosan végrehajtható, illetve például figyelembe vehetőek az EUROCODE szabvány által meghatározott, kihajlása és horpadásra vonatkozó feltételek is, valamint a cég különböző gyártástechnológiára és gyárthatóságra vonatkozó korlátait egyaránt.

Bár sok vázvonal- vagy „csontváz” detektáló algoritmus létezik, azonban ezek nem kifejezetten az optimálási feladatok végeredményének feldolgozására lettek kifejlesztve, ezért az alábbiakban ismertetésre kerül egy bitminta-primitív szűrők segítségével működő

* *gépészmérnök MSc hallgató, Miskolci Egyetem*
** *egyetemi tanár, Miskolci Egyetem, Anyagmozgatási és Logisztikai Tanszék*

detektáló algoritmus (ld. 1. ábra), mely a tesztek folyamán 100%-os megbízhatóságot mutatott.



1. ábra. Tervezés folyamata

Az alábbiakban néhány, a későbbiekben alkalmazásra kerülő fogalmat definiálunk.

Topológiai szerkezet (TD – Topology Design): A topológiai optimálás végeredménye, ami tervezési tartományon belül értelmezett sűrűség-értékekből áll (azaz az optimált változók értékeiből; SIMP esetén ezek a $[0,1]$ vagy $[x_{\min},1]$ tartományban található értékek [10,11]; ESO/BESO esetén ezek a diszkrét $0.0/1.0$ vagy $x_{\min}/1$ értékek hard- illetve soft-kill eljárás esetén [12,13,14]). A diszkretizáció történhet strukturált vagy strukturálatlan véges-elemes hálón, esetleg egyéb eljárásokkal is.

Csomópont: Két vagy több rúd találkozása.

Végleges csomópont: Olyan rudak kapcsolódási pontja, amelynek a végleges topológiában is léteznie kell.

Evidens csomópont: Három vagy több rúd közötti csatlakozási pont. Egyben végleges csomópont is.

Átmeneti csomópont: Pontosan két rúd közötti kapcsolódási pont. Ez a csomópont jelenthet egy átmeneti kapcsolatot a detektálás folyamatában, vagy egy végleges csomópontot amennyiben a rudak szöget bezáróan kapcsolódnak.

2. ábra. SIMP megoldása (TD: 160x40)

3. ábra. RTD (TD: 960x160)

3. DETEKTÁLÓ ALGORITMUS FELÉPÍTÉSE

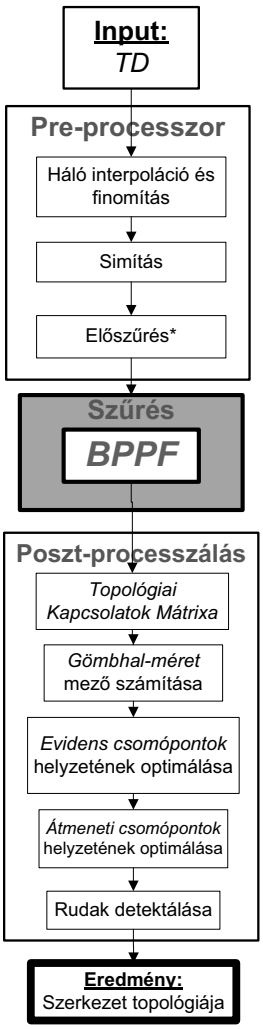
A detektáló algoritmus többszintű, bemeneti adata egy TD, és az alábbi lépésekre osztható fel (ld. 4. ábra):

1. lépés: A véges-elemes hálón alapuló *TD* alapján a *BPPF* számára is értelmezhető adat létrehozása.

a, Háló interpoláció és simítás: Ezen lépés célja az *RTD* létrehozása, mely az alábbiak szerint lesz definiálva:

Finomított Topológiai Szerkezet (RTD - Refined Topology Design): Strukturált diszkretizált négyzetes (3D esetén kocka alapú) háló, mely a sűrűségértékeket tárolja. A *TD* értékeinek interpolációjával állítható elő. Amennyiben az eredeti *TD* is ilyen hálóval került kiszámításra, akkor az *RTD* megegyezhet a *TD*-vel, ellenkező esetben két eljárás használata javasolt az interpolációhoz.

Az első esetében a nagyon finom struktúrált négyzetes (2D)/kocka alapú (3D) hálót az eredeti hálón belül, megfelelő nagyszámú pont esetén történő mintavételezéssel, míg a második esetben a cellánkénti sűrűségértékek csomópontokra történő átszámításával, majd ezek alapján az eredeti véges-elem háló alakfüggvényeivel történő interpolációval állítható elő.



4. ábra. Algoritmus felépítése

b, Simítás: Mivel a hálóinterpoláció eredményeként egyes esetekben az eredeti cellák különálló szerkezeti elemekként, blokkonként jelennek meg, ezért ajánlott a *TO*-ban alkalmazott hálófüggetlenségi szűrőhöz [10,14,15] hasonló szűrőt alkalmazni a szerkezetre $r_s = (0.5-1.0)r_{\min}$ sugárral, ahol $r_{\min}$ a *TO*-ban alkalmazott szűrő sugara.

A hálófinomítási tényező jelölése Ref_{mesh}, ami azt mutatja meg, hogy hányszor finomabb az interpolált háló az eredetinél. Ajánlott értékei *2-4*.

$$Ref_{mesh} = \sqrt[x]{\frac{\text{új háló elemeinek száma}}{\text{régi háló elemeinek száma}}} \quad \text{, ahol } \begin{matrix} x=2 \text{ 2D-re} \\ x=3 \text{ 3D-re} \end{matrix} \quad (1)$$

c, Előszűrés: Az előszűrés egy meghatározott *BPPF* alkalmazását jelenti, amely tovább finomítja az *RTD*

egyes részleteit, így a kapott nagyfelbontású, jó minőségű kép akár más algoritmusokhoz is alkalmazható (ld. 5. ábra).

5. ábra. RTD (simítással)

4. BPPF ÁLTALÁNOS MEGFOGALMAZÁSA

Az alábbiakban ismertetésre kerül a *BPPF* felépítése és legfontosabb fogalmai 3D-ra. A 2D-s megközelítés nem kerül ismertetésre, azonban minden ismertetett képlet alkalmazható 2D-re is gond nélkül, amennyiben a 3. dimenziót mindenütt figyelmen kívül hagyjuk benne. Bár hasonló középvonal-detektáló algoritmusok léteznek [16], azonban ezek eltérő matematikai feltételekkel közelítik meg a feladatot, és nem tartalmaznak kiegészítő algoritmusokat az eredmények további javítására, illetve a szűrés nem szabályozható akár futásidőben. Az alábbiakban a *BPPF*-hez szükséges további definíciók következnek.

Közbenső Topológiai Szerkezet (ITD - Intermediate Topology Design): Amennyiben bármilyen szűrő (sikeresen) alkalmazásra került az *RTD*-re, akkor az egy *ITD* lesz. Más néven ez lesz az alap voxelmező (mivel minden cella 0 vagy 1 értéket tárol benne).

Voxel cella: Az *RTD* vagy *ITD* egyetlen cellája. Egy voxel jelölése $\lfloor l,m,n \rfloor$, azaz ez az *l*-dik cella az *x* irányban, valamint az *m*-dik cella az *y* irányban illetve az *n*-dik cella a *z* irányban.

Végleges Topológiai szerkezet (TDR - Topology Design Results): A szűrési folyamat végeredménye, ahol minden egyes voxel csak és kizárólagosan két szomszédos voxellel rendelkezik, amennyiben az nem egy *evidens csomópontot* képvisel.

Érték operátor: Egyetlen cella értékét mutatja meg: $\hat{f}\lfloor l,m,n \rfloor$. Üres cellák esetén ez $\hat{f}\lfloor l,m,n \rfloor = 0$, míg tömör cellák esetén $\hat{f}\lfloor l,m,n \rfloor = 1$ értéket ad vissza a *TDR* vagy *ITD* egyetlen cellájára.

Általános szűrő: Bármely r_x x r_y x r_z méretű téglalap alakú terület.

Szimmetrikus szűrő: Bármely általános szűrő egyben szimmetrikus szűrő is, ha $r_x = r_y = r_z$ teljesül.

Szűrő cella: Egy szűrő egyetlen celláját azonosítja: $\langle i,j,k \rangle$, ha $1 \le i \le r_x$ és $1 \le j \le r_y$, $1 \le k \le r_z$.

Érték operátor (szűrő esetén): Egyetlen szűrő cella értékét mutatja meg: $\hat{f}\langle i,j,k \rangle$. Az implementáció során egy három-értékű reprezentáció került megvalósításra, ahol egy általános szűrő bármely celláját üres szűrő cellának nevezzük, amennyiben $\hat{f}\langle i,j,k \rangle = 0$, tömör szűrő cellának, amennyiben $\hat{f}\langle i,j,k \rangle = 1$ és detektor szűrő cellának, ha $\hat{f}\langle i,j,k \rangle = -1$.

Egyfókuszú szűrő: Bármely általános szűrő, amelyre teljesül, hogy $\hat{f}\langle i_1,j_1,k_1 \rangle = \hat{f}\langle i_2,j_2,k_2 \rangle = -1$ akkor és csakis akkor, ha $i_1 = i_2$, $j_1 = j_2$, és $k_1 = k_2$ azaz egyetlen detektor szűrő cellát tartalmaz.

Aktív szűrő: Bármely szűrő aktív szűrő a $\lfloor l,m,n \rfloor$ pozíciónál, ha $\hat{f}\lfloor l+i,m+j,n+k \rfloor = 0$ akkor és csakis akkor igaz, ha $\hat{f}\langle i,j,k \rangle = 0$, $1 \le i \le r_x$, $1 \le j \le r_y$, $1 \le k \le r_z$. Az aktív szűrőket az aktiválási operátor 1 értékkel jelzi: $\hat{A}_{filter}\lfloor l,m,n \rfloor = 1$ (egyébként 0 értéket ad vissza). Amennyiben az aktiválási operátor feltétele teljesül, akkor a szűrő hatása az alap voxelmezőre $\hat{f}\lfloor l+i,m+j,n+k \rfloor := 0$ lesz bármely szűrőcellára, ahol $\hat{f}\langle i,j,k \rangle = -1$ teljesül.

Szomszédossági operátor: Bármely két szűrő cella szomszédos akkor és csakis akkor, ha van egy közös lapjuk. Ez gyakran lesz közvetlen kapcsolatként is megnevezve. Ezt az állapotot a szomszédossági operátor 1-es értékkel fogja jelezni: $\hat{N}\langle i_1,j_1,k_1 \rangle \langle i_2,j_2,k_2 \rangle = 1$ (egyébként 0 értéket ad vissza).

Kapcsolódási operátor: Bármely két cella kapcsolatban áll egymással egy úton keresztül, ha olyan megfelelő indexpárokból álló sorozat létezik, melyben minden indexpár által reprezentált cella nem üres cella, és az egymás utáni indexek szomszédos cellákat jelölnek. A kapcsolódást a kapcsolódási operátor 1-es értékkel fogja jelezni: $\hat{C}\langle i_1,j_1,k_1 \rangle \langle i_2,j_2,k_2 \rangle = 1$ (egyébként 0 értéket ad vissza).

Érvényességi operátor: Az érvényességi operátor 1-es értéke fogja jelezni $\check{V}[filtername] = 1$ (egyébként 0 értéket ad vissza) annak a szükséges és elégséges feltételét, hogy ne csökkentsük a csomópontok és rudak számát a topológiában. Ennek megfelelően csak azok a szűrők megengedettek, amelyek nem tudnak letörölni egyetlen olyan pixelt sem, amelyek lehetnének az egyszerűsített szerkezet vázvonalának részei (azaz a detektáló szűrő cella nem lehet a szűrő szélén, illetve csak olyan pixelek törölhetőek, amelyek tömör pixelek olyan csoportjait kötik össze, melyek a detektáló szűrő cellát nem tartalmazó úton is kapcsolódnak egymáshoz.

Ezek a feltételek azonban nem elégségek ahhoz, hogy az algoritmus ne hozzon létre lokális hibákat, hurkokat, azaz szükséges definiálni az ehhez szükséges feltételeket is.

Konzisztens ITD: Egy olyan *ITD*, amelyben nem léteznek olyan élek vagy csúcsok, amelyek csak két olyan voxel cellához tartoznának, amelyek nem szomszédos cellák. Ezt az állapotot a konzisztens érvényességi operátor $\check{V}^C\lceil filtername \rceil = 1$ értékkel jelzi. Azokat a szűrőket, amelyek kielégítik ezt a feltételt, konzisztens szűrőknek fogjuk nevezni.

Bár a szerkezet immár konzisztens marad végig, de a csomópontok száma még mindig nőhet. Ennek elkerülése végett bevezetésre kerül az inverz szerkezet fogalma. Amikor a szerkezeten a szűrést hajtjuk végre,

valójában egy másik – rejtett – szerkezet is létezik, amely a mi szerkezetünk inverze (azaz az üres cellákból áll: $\hat{f}\langle i,j,k\rangle = 0$). Ha megvizsgáljuk ezt a szerkezetet, nyilvánvalóvá válik, hogy az inverz szerkezet topológiáját is sértetlenül kell hagynunk, akárcsak a konzisztenciáját. Azokat a szűrőket, amelyek kielégítik azt a feltételt, hogy az inverzük is konzisztens marad, inverzen konzisztens szűrőknek nevezzük, és ezt az állapotot az inverzen konzisztens érvényességi operátor: $\check{V}^{IC}\lceil filtername\rceil = 1$ jelöli.

Emellett belátható, hogy ha törlünk egy elemet a szerkezetből, akkor egy elemet hozzáadunk annak inverzéhez. Amennyiben ez az új elem két, korábban egymással nem kapcsolódó részét köti össze az inverz szerkezetnek, akkor abban egy új rúd jött létre, azaz a topológia sérült. Ez utóbbi feltételt kielégítő szűrőket fogjuk inverzen érvényes szűrőknek nevezni, melyeknek ki kell elégíteniük az inverz érvényességi operátort: $\check{V}^{I}\lceil filtername\rceil = 1$. Matematikailag megfogalmazva $\check{V}^{I}\lceil filtername\rceil = 0$ bármely olyan szűrőre, amely két egymáshoz nem kapcsolódó üres régiót összekötne az aktiválás után.

Ha megpróbáljuk meghatározni az összes lehetséges érvényes detektáló szűrőt, akkor egy erősen korlátozott tagszámú csoportot kapunk – amelyek forgatással vagy tükrözéssel nem transzformálhatóak egymásba-, ezt a csoportot fogjuk bitminta-primitíveknek nevezni. 2D esetén 9 primitív elegendő a szűrés elvégzéséhez, 3D esetében azonban jóval nagyobb számú létezik, ezért ebben az esetben bitminta-primitív csoportokat definiálunk, melyeket különböző tulajdonságaik alapján azonosíthatunk (pl. tömör cellák száma, stb.). Ezeket a csoportokat adhatjuk meg a programnak megfelelő szekvencia-operátorok segítségével, melyek azt határozzák meg, hogyan és milyen sorrendben kerülnek alkalmazásra az egyes szűrők (pl. az alábbi megvalósításban $\tilde{Z}$, $\tilde{B}$, $\tilde{F}$, $\tilde{M}$, $\tilde{S}$, $\tilde{D}$ operátorok).

5. POSZT-PROCESSZÁLÁS

A poszt-processzálás feladata nem más, mint a legkisebb olyan *TCM* (lásd később) megalkotása, mely megfelelően le tudja írni a szerkezet topológiáját. Bár a szerkezet ‘csontváza’ sikeresen detektálva van, de ennek ellenére az eredmények ebben a formában még nem használhatóak. Az evidens csomópontok azonnal felismerhetőek, de azok a végleges csomópontok, amelyek szög alatt csatlakozó rudakat kötnek össze, még nem, illetve a kötések típusa sem. Az alábbiakban röviden ismertetésre kerülnek az algoritmus egyes része.

a, Topológiai Kapcsolatok Mátrixának előállítása (a továbbiakban TCM - Topology Connectivity Matrix): Egy olyan mátrix, amely a csomópontok számát, és azok közötti kapcsolatokat (rudak) határozzák meg. Amennyiben a csomópontok számát *n*-nel jelöljük, akkor ez egy *n x n* méretű mátrix. Amennyiben az *i*-dik csomópont kapcsolódik a *j*-dik csomóponthoz, akkor ez fordítva is igaz, azaz $t_{ij} = t_{ji} = 1$, tehát a mátrix szimmetrikus: $\mathbf{T} = \mathbf{T}^T$.

Egyenes középvonalú tartók esetén t_{ij} nem nulla értéke jelzi a kapcsolatot, egyébként pedig ez az érték határozza meg a gerenda típusát.

Erősen ajánlott számítástechnikai okokból egy ’sparse’ mátrixban tárolni az adatokat, mivel ez a mátrix egy ritka mátrix. Az egyes csomópontok pozícióját a $\underline{\mathbf{p}}$ –vel jelölt pozíció vektor fogja tárolni. Egyetlen csomópont pozíciója a $\underline{\mathbf{p}}_i$ helyen található, és az egyes komponenseket pedig $p_{i,x}$, $p_{i,y}$ jelöli.

Az előállítás során jellemzően $Step_{mov} = (0.4 - 0.7) \cdot Ref_{mesh} \cdot r_{\min}$ számú lépésenként a voxelekből álló csontváz alapján létrehozunk egy nagyfelbontású, nagy elemszámú *TCM*-et, későbbi feldolgozás céljából.

b, Gömbhal operátor számítása: A gömbhal operátor elnevezés egy egyszerű hasonlatból származik. A szerkezet minden egyes pontjába gömbhalakat helyezünk el, majd azok akkorára fúvódnak fel, amíg a szerkezet falához nem érnek. A gömbhal operátor értéke (ld. 6. ábra) tulajdonképpen egy adott pontba beírható legnagyobb olyan gömb térfogatát jelenti, melynek felszíne nem metszi a szerkezet határvonalát.

6. ábra. Gömbhal operátor (nagyobb érték sötétebb)

c, Lokális csomóponti pozíció optimálás (LPPO - Local point position optimization): Ezen optimálás alapötlete, hogy a *BPPF* eredményeként lehetséges, hogy a *TCM* csomópontjai nem optimális helyzetben (értsd nem pontosan a középvonalon) helyezkednek el, de közel vannak hozzá. Ebben az esetben egy kis korrekció elvégzésével egy sokkal pontosabb megoldást érhetünk el. Az optimális pozíciót a gömbhal operátor magasabb értékei jelzik. Az *LPPO* folyamat a csomópontok közvetlen környezetét fogja átvizsgálni olyan pozíciók után kutatva, ahol a gömbhal operátor értéke magasabb, azaz közelebb található a középvonalhoz. Amennyiben azonban több csomópont is a skalármező egyik maximumának közelében található, akkor előfordulhat, hogy az összes csomópont lépésről lépésre a maximumhoz közelítene, így hibás eredményt hozva létre. Ennek elkerülésére bevezethető egy további paraméter, $\alpha_{\max}$, azaz a tiltott szögtartomány. Így egy optimálási feltétellé tesszük azt is, hogy egyik csomópont sem mozoghat olyan irányokba, ami közel van egy másik, létező kapcsolat

irányához, azaz megtiltunk minden olyan mozgást, amely egy már létező kötés irányában történne.

Maximáljuk:

$$\hat{P}\left\lfloor p_{i,x}+d_x, p_{i,y}+d_y, p_{i,z}+d_z \right\rfloor \rightarrow \max$$

Úgy, hogy $\theta_j \le \alpha_{\max}$, ahol

$$\theta_j = \arccos \frac{\left(d_x\mathbf{e_x}+d_y\mathbf{e_y}+d_z\mathbf{e}_z\right)\cdot\left(\mathbf{p_i}-\mathbf{p_j}\right)}{\left|d_x\mathbf{e_x}+d_y\mathbf{e_y}+d_z\mathbf{e}_z\right|\cdot\left|\mathbf{p_i}-\mathbf{p_j}\right|}$$, ha $t_{ij}=1$.

(a lehetséges mozgás iránya és a csomópontot egy másik csomóponttal összekötő irány által bezárt szög)

ahol, $0 \le d_x, d_y, d_z \le D_{\max}$

Az új pozíció pedig $\mathbf{p_i^{new}} = \mathbf{p_i^{old}} + d_x\mathbf{e_x} + d_y\mathbf{e_y} + d_z\mathbf{e}_z$.

Általában $D_{\max} = (0.8-1.1)\cdot Ref_{mesh}$ érték egy jó választás a folyamathoz. A javasolt paraméterek a szögekre $\alpha_{\max} = 6°$ az evidens csomópontokra, és $\alpha_{\max} = 12°$ az átmeneti csomópontokra. A teljes optimálási folyamatot addig ismételjük, amíg a pozíciók nem konvergálnak.

d, Szerkezeti elemek detektálása: Az algoritmus legutolsó része a konkrét szerkezetei elemek detektálása, mivel ahhoz, hogy egy szerkezetet pontosan detektálhassunk, szükséges információkkal rendelkeznünk annak céljáról, méreteiről, és gyártástechnológiájáról egyaránt.

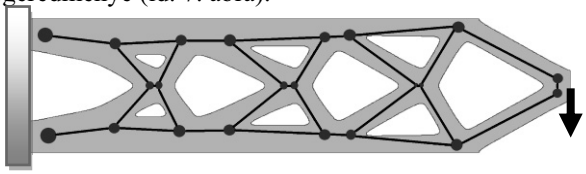
A szoftver egy adatbázisban tárolhatja a különféle rudakra és kötésekre vonatkozó feltételeket, és a tervező egyszerűen kiválaszthatja, mely szerkezeti elemekkel (egyenes vagy ívelt) szeretne dolgozni a poszt-processzálás folyamán. Az eredmények bemutatása során egy egyszerű, de meglepően hatékony eljárás került implementálásra egyenes rudak detektálására.

6. EREDMÉNYEK

Az alábbiakban bemutatásra kerül néhány optimálási feladat eredménye az algoritmus segítségével feldolgozva. A topológiák BESO [12,13,14], SIMP [3,10], illetve QSQF [17] algoritmus segítségével kerültek előállításra.

6.1 Egyik végén befalazott, másik végén hajlított rácsos tartó – 2D

Az alábbiakban a cikk folyamán már többször is bemutatott példán kerül ismertetésre az algoritmus végeredménye (ld. 7. ábra).

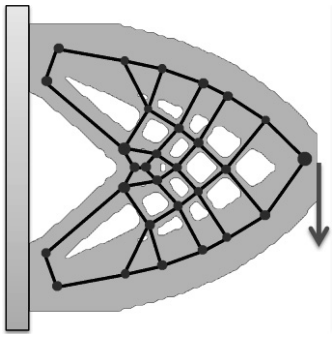


7. ábra. Rácsos tartó

A nagyobb pontok a megfogás helyét, a kisebbek az evidens csomópontokat, míg a sötét vonalak a rudakat jelölik. A háttérben halvány szürkével még látható az eredeti szerkezet.

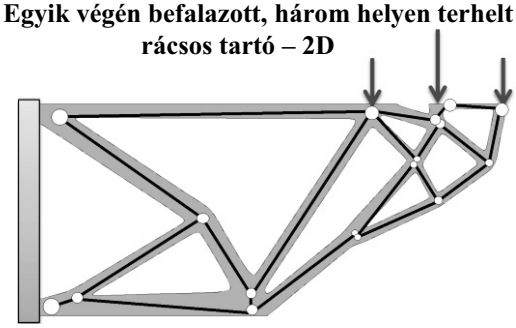
6.2 Egyik végén befalazott, másik végén hajlított rácsos tartó, eltérő méretarányokkal – 2D

Az alábbiakban az előzőhöz hasonló, azonban eltérő méretarányokkal rendelkező, és nagyobb térfogataránnyú példa kerül bemutatásra (ld. *8. ábra*).



8. ábra. Rácsos tartó

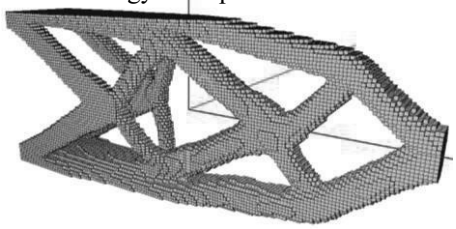
6.3 Egyik végén befalazott, három helyen terhelt rácsos tartó – 2D



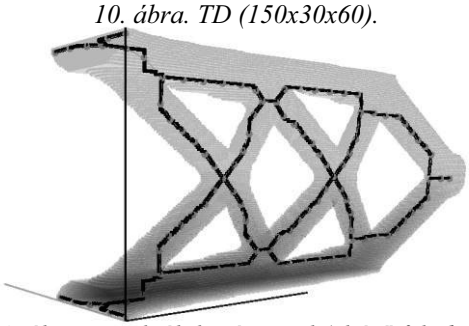
9. ábra. Rácsos tartó.

6.4 Egyik végén befalazott, másik végén hajlított rácsos tartó – 3D

Az alábbiakban egy 3D-s példa kerül bemutatásra.



10. ábra. TD (150x30x60).



11. ábra. Detektált középvonal (eltérő feladat).

7. ÖSSZEFOGLALÁS

A cikkben javasolt eljárás nem csak gyors és megbízható, de viszonylag egyszerűen implementálható is, a tesztek során az eredeti *TD* méretétől függően 1-60 másodperc alatt elkészült a számítással (2D) egy Intel® Core i5™ 760 processzor egyetlen magján (3.2 GHz). Az algoritmus nagyon könnyen párhuzamosítható mind a CPU-n, mint a GPU-n a nagyszámú primitívnek hála.

A fejlesztés következő szakasza a méretoptimálási modul, az *LPPO* egy továbbfejlesztett változatának létrehozása, valamint végül, de nem utolsó sorban egy olyan adatbázis létrehozására fog irányulni, amely sok, mérnöki szempontból is jelentőséggel bíró kötés- és rúdtípus kritériumait fogja tartalmazni. Bár az alábbi munkában a *BPPF* kifejezetten a középvonalak detektálására került alkalmazásra, de a képességeinek és a lehetőségek széles tárházának hála (pl. szűrők, szűrő méret, *LPPO* paraméterek, stb.) az eljárás nagy valószínűséggel kiterjeszthető nem csak középvonal, de például lemezek, felületek, és egyéb mérnöki szempontból fontos „feature" csoportok detektálásra is, vagy akár a képfelismerés területeire is.

8. KÖSZÖNETNYILVÁNÍTÁS

A bemutatott kutató munka a TÁMOP-4.2.1.B-10/2/KONV-2010-0001 jelű projekt részeként az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg, valamint az Országos Tudományos Kutatási Alap OTKA T 75678 támogatásával.

9. IRODALOM

[1] Liu, Z. Gao, Q., Zhang, P., Xuan, M. and Wu, Y.: Topology optimization of fluid channels with flow rate equality constraints. Structural and Multidisciplinary Optimization, Volume 44, Number 1, 31-37. 2011.

[2] Rozvany, G.I.N.: Domain augmentation and reduction in structural topology optimization. Structural and Multidisciplinary Optimization, Volume 44, Number 4, 141-158. 2011.

[3] Bendsoe, M. P., Sigmund, O.: Topology Optimization – Theory, Methods and Applications. Chapter 4, 16-68. Springer, 1995.

[4] Bendsoe, M. P., Ben-Tal, A., and Zowe, J.: Optimization methods for truss geometry and topology design. Structural and Multidisciplinary Optimization, Volume 7, Number 3, 141-158. 1994.

[5] Nursel Öztürk, Ali R. Yildiz, Necmettin Kaya, Ferruh Öztürk: Neuro-Genetic Design Optimization Framework to Support the Integrated Robust Design Optimization Process in CE. Concurrent Engineering, Volume 14, Number 1, 5-16, March 2006.

[6] Yildiz, A.R., Öztürk, N., Kaya, N., Öztürk, F.: Integrated optimal topology design and shape optimization using neural networks. Structural and Multidisciplinary Optimization, Volume 25, Issue 4, 251-260, 2003.

[7] William Robert Blattman: Generating CAD Parametric Features based on Topology Optimizations. Department of Mechanical Engineering, Brigham Young University, MSc Thesis. 2008.

[8] Ulrich Bauer, Konrad Polthier: Detection of Planar Regions in Volume Data for Topology Optimization. Advances in Geometric Modeling and Processing, 119-126, 2008.

[9] Farkas,J. ,Jármai, K.: Design and optimization of metal structures. 328 p. ISBN: 978-1-904275-29-9. Horwood Publishers, Chichester, UK, 2008.

[10] Sigmund, O.: A 99 line topology optimization code written in Matlab. Structural and Multidisciplinary Optimization. Volume 21, Number 2, 120-127. 2001.

[11] Zhou, M., Rozvany, G.I.N.: The COC algorithm, part II: Topological, geometry and generalized shape optimization. Computer Methods in Applied Mechanics and Engineering, 89. 1991.

[12] Zhou, M., Rozvany, G. I., N.: On the validity of ESO type methods in topology optimization. Structural and Multidisciplinary Optimization, Volume 21, Number 1, 80-83, 2001.

[13] Querin, O.M., Steven G.P., Xie, Y. M.: Evolutionary structural optimisation (ESO) using a bidirectional algorithm. Engineering Computations, Volume 15, Number 8, 1031-1048. 1998.

[14] Huang, X., Xie, Y. M.: Evolutionary Topology Optimization of continuum Structures – Methods and Applications. Wiley, 2010.

[15] Sigmund, O., Petersson, J.: Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh independencies, and local minima. Structural and Multidisciplinary Optimization, Volume 16, Number 1, 68-75. 1998.

[16] Louisa Lam, Seong-Whan Lee, Ching Y. Suen: Thinning Methodologies – A Comprehensive Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 14, Number 9, 869-885, September 1992.

[17] Daróczy, L., Jármai K.: New method for topology optimization of truss structures. GÉP folyóirat, LXII. évfolyam, I. Kötet, 29-34. Miskolc, 2011. (in Hungarian)