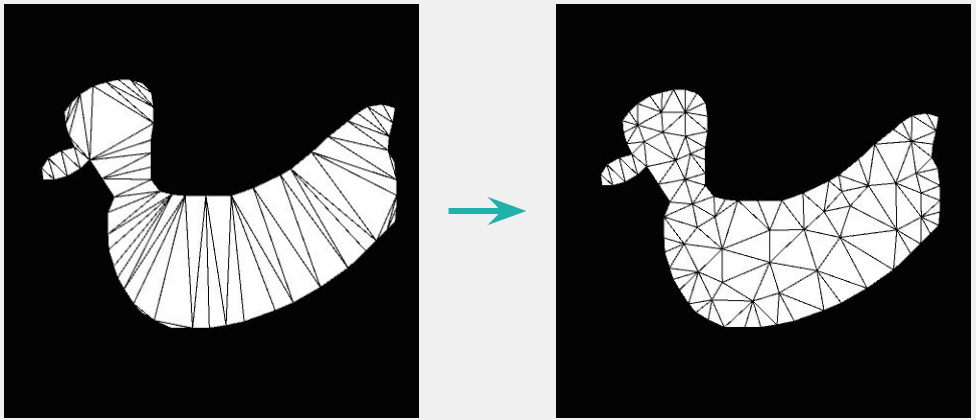
Ruppert’s algorithm

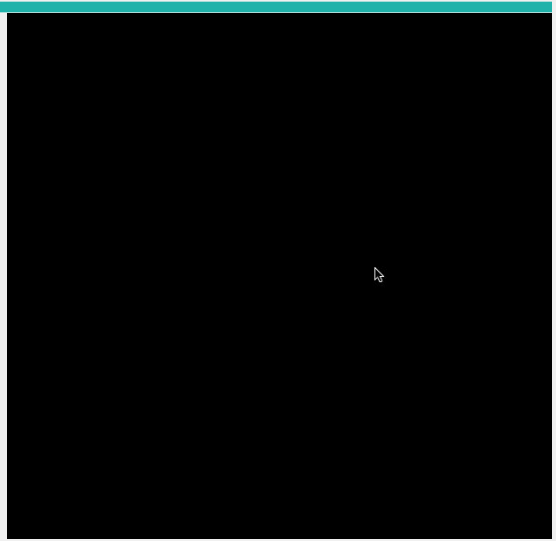
Constrained Delaunay Triangulation

3D számítógépes geometria 2

Motiváció

- Adott határgörbe egyenletes kitöltése
- Degenerált háromszögek kiküszöbölése
- További kényszerek mellett
- Véges elemes módszer
- 2D domain 3D-be leképzés

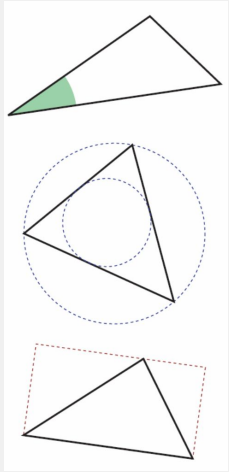




Forrás: https://github.com/mattatz/unity-triangulation2D

Háromszöghálók jellemzése

Izotrópia: „egyenletes” élhosszak és „egyenletes” szögek

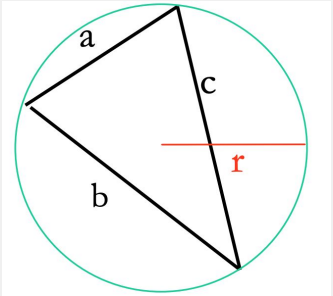


Mértékek:

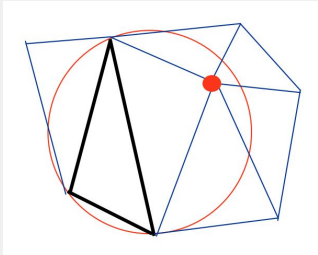
- legkisebb szög
- a körülírt kör és a belső érintőkör sugarának aránya
- a leghosszabb oldal és a magasság aránya
- a legrövidebb oldal és a körülírható kör sugarának aránya

$$\frac{min(a,b,c)}{r}$$

maximális, ha a=b=c, $\sqrt{3}$

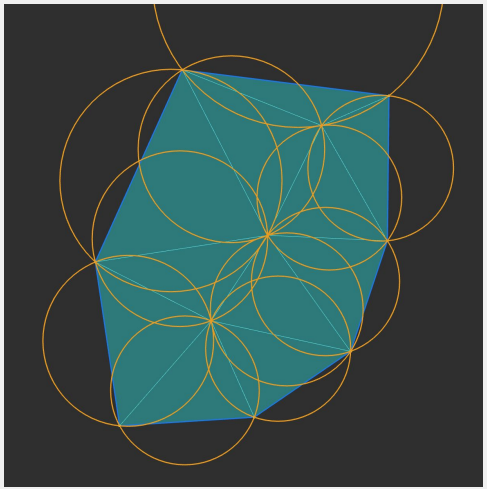


Delaunay háromszögelés (DT)



- A körülírható körökben nincs más pont
- Egyértelmű, ha nincsenek környégyszögek
- Egyenletes háromszögeloszlás

- Maximalizálja a legkisebb szöget
- Minimalizálja a legnagyobb körülírható kör sugarát
- Minimalizálja a beírható körök sugarainak összegét

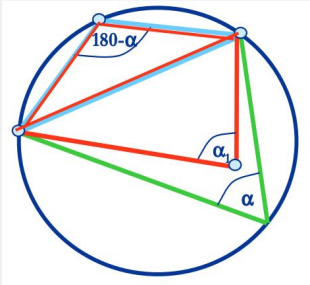


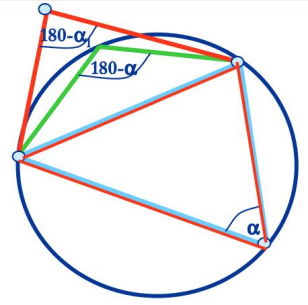
Élcsere algoritmus

- Input: tetszőleges háromszögelés
- konvex négyszögek vizsgálata
- ha szükséges, lokális élcsere (edge flipping)
- az eljárás mindig terminál – $O(n^2)$ lépés

Környégyszög kritérium

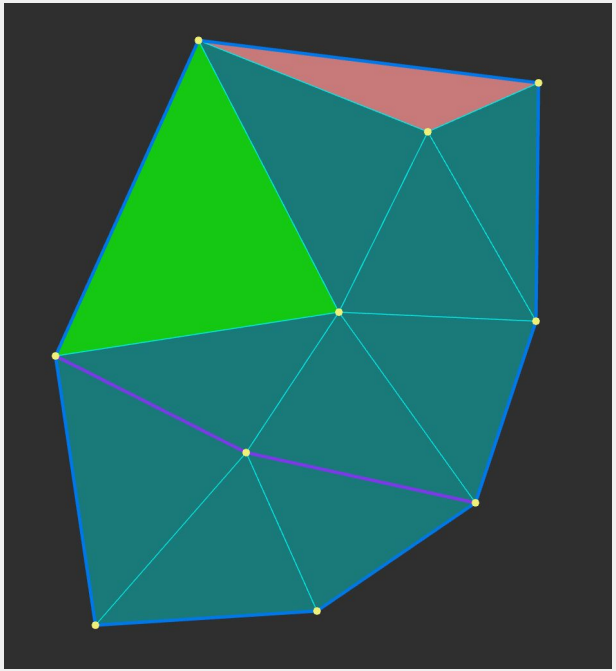
- szemben levő szögek összege: 180
- első eset: $\alpha_1 + (180 - \alpha) > 180 \leftrightarrow$ rossz átló
- második eset: $\alpha + (180 - \alpha_1) < 180 \leftrightarrow$ jó átló
- mindig a nagyobb szögpár csúcsait kell összekötni!





Jelölés

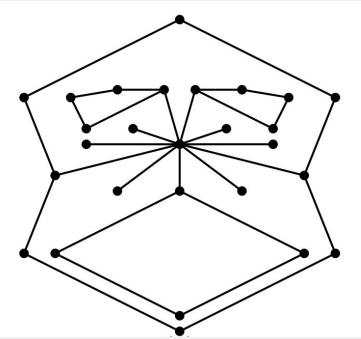
- Legkisebb szögű háromszög
- Legnagyobb területű háromszög
- Rögzített él
- Határvonal

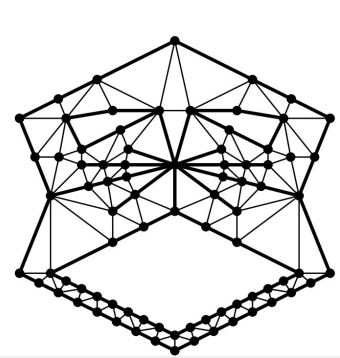


Ruppert’s algoritmus

- Bemenet: Planar Straight Line Graph

- Kimenet:
  Delaunay háromszögelés kényszerekkel
  - Minimális szög korlát
  - Maximális terület korlát
  - Rögzített élek megtartása

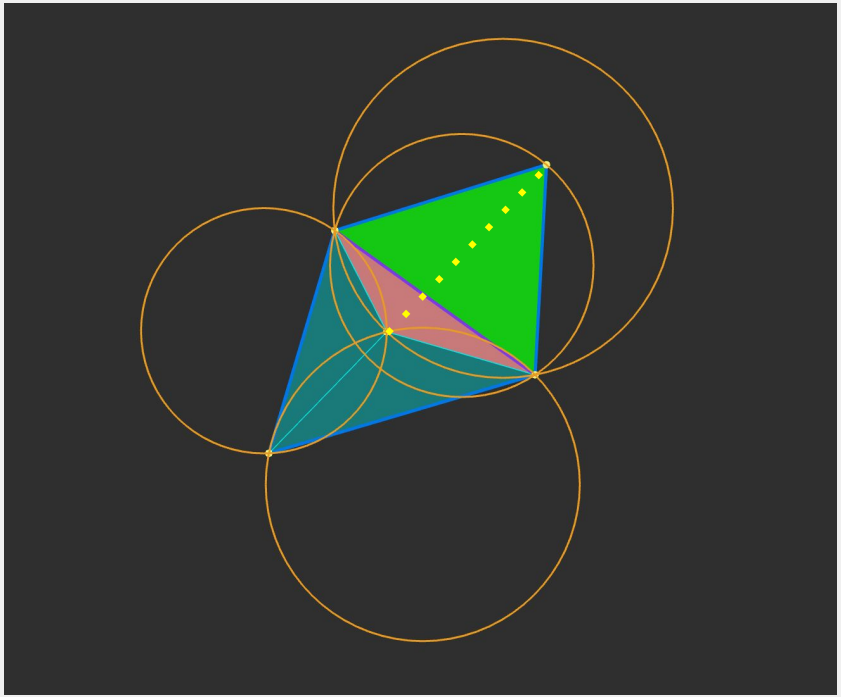




Constrained Delaunay Triangulation

- Rögzített élek a háromszögelésben
- Nem Delaunay háromszögelés

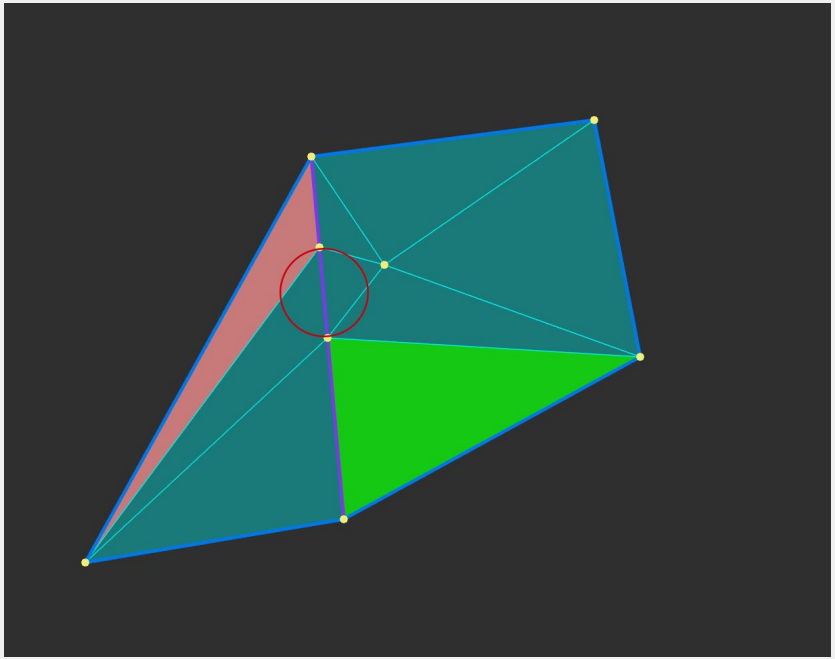
- A körülírható körökben nincs más **látható** pont



Beavatkozó csúcsok (encroaching)

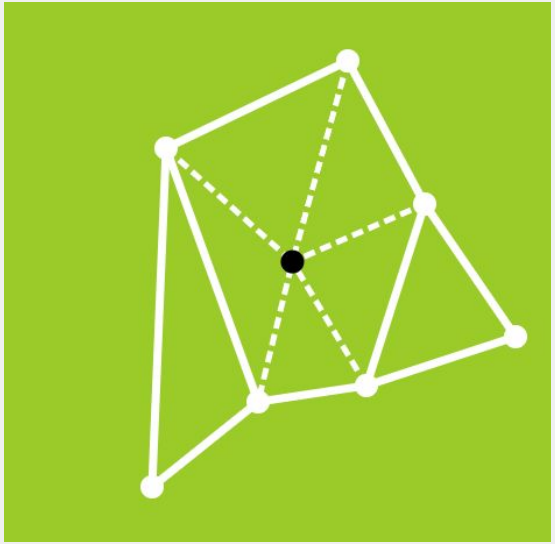
Szakasz befoglaló körébe belelóg másik csúcs:

- Szakasz felezése
- Újraháromszögelés



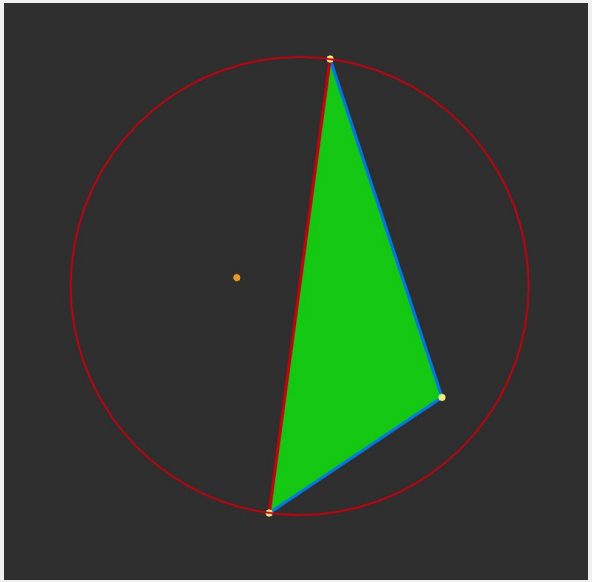
Rossz háromszögek

- Csúcs beszúrása a háromszög körülírható körének középpontjába
- Ha beavatkozik rögzített élbe
  - Él felezése
- Különben
  - Csúcs beszúrása
  - Tartalmazó háromszög felosztása
  - Újra háromszögelés



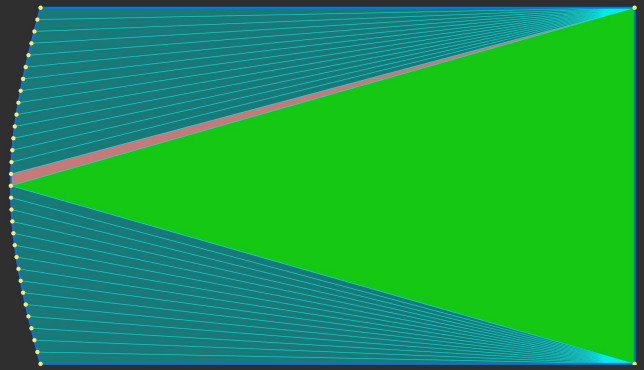
Ruppert’s algoritmus

1. Beavatkozott (encroached) élek felezése
2. Rossz háromszögek megszűntetése
3. Goto 1.

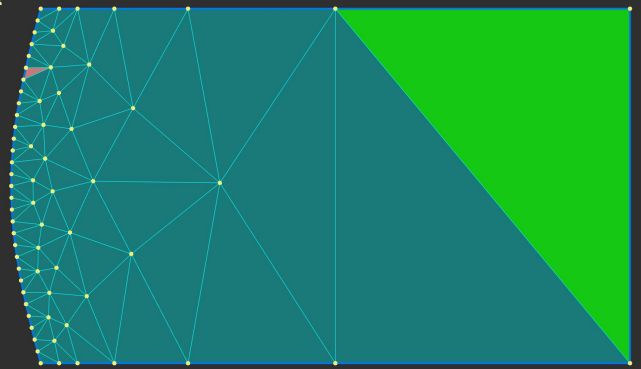


- Ha nincsenek beavatkozó csúcsok, a beillesztendő csúcsok mindig az alakzaton belül találhatóak

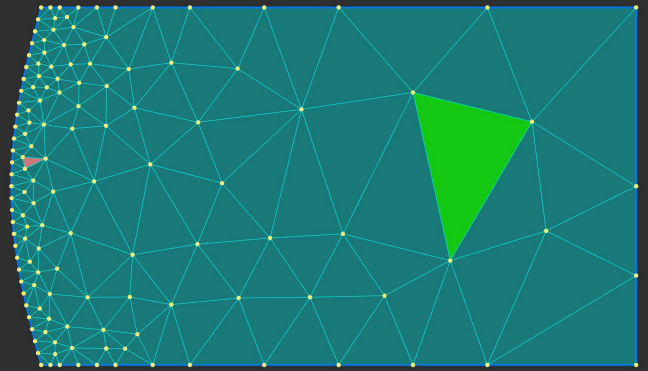
Szög-kényszerek



31 ▲

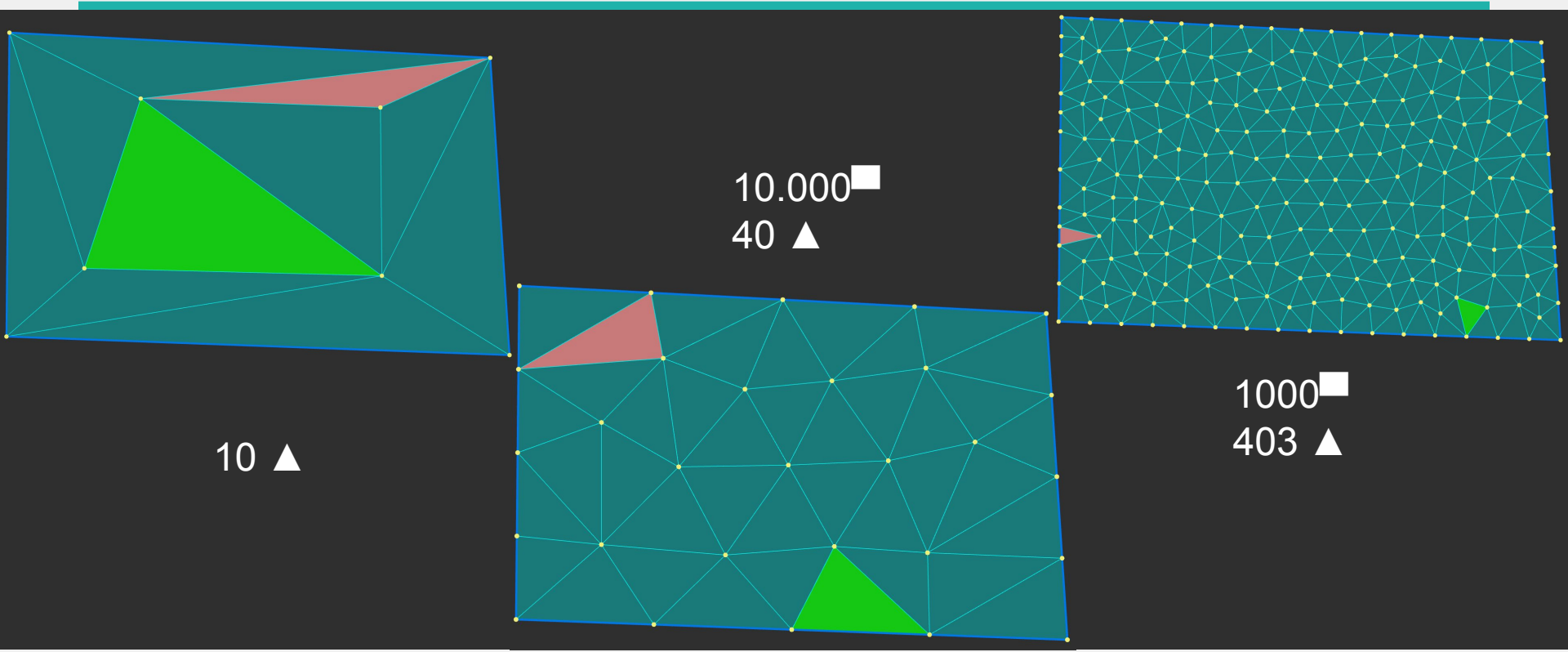


22°
95 ▲

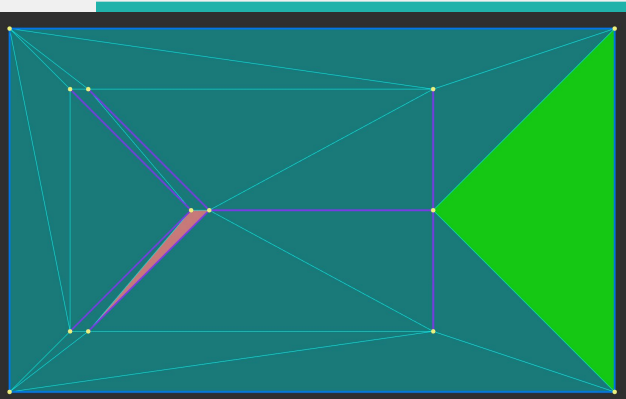


30°
216 ▲

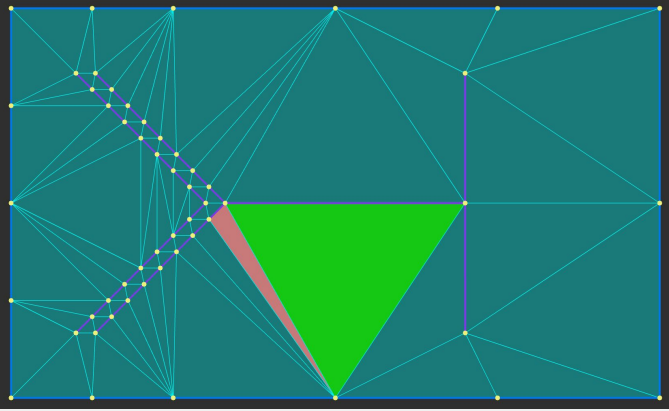
Terület-kényszer



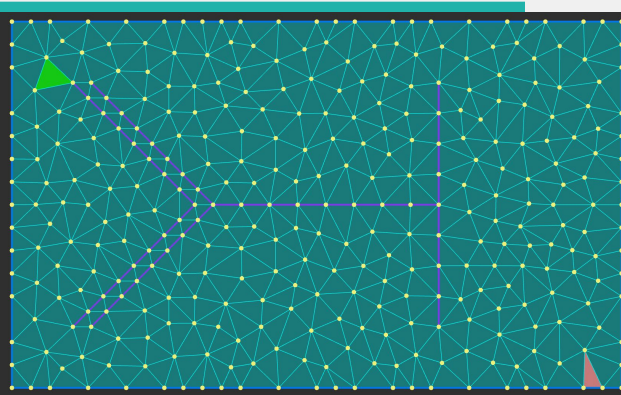
Rögzített élek



20 ▲



88 ▲



25°
1000
612 ▲

Leállási feltétel

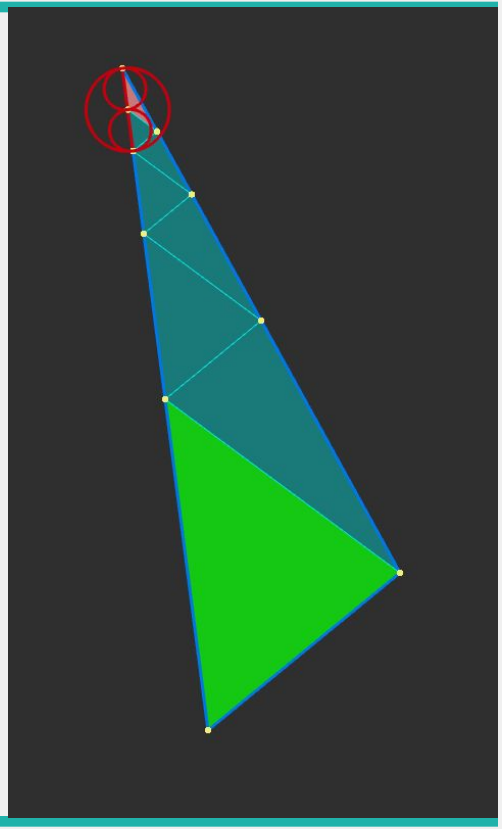
Ideális feltétel:

$$\frac{min(a,b,c)}{r} \geq \sqrt{2}$$

Rögzített élek közötti szög $\geq 60^\circ$

Futásidő: $\Theta(h^2)$

h - kimeneti háromszögelés mérete

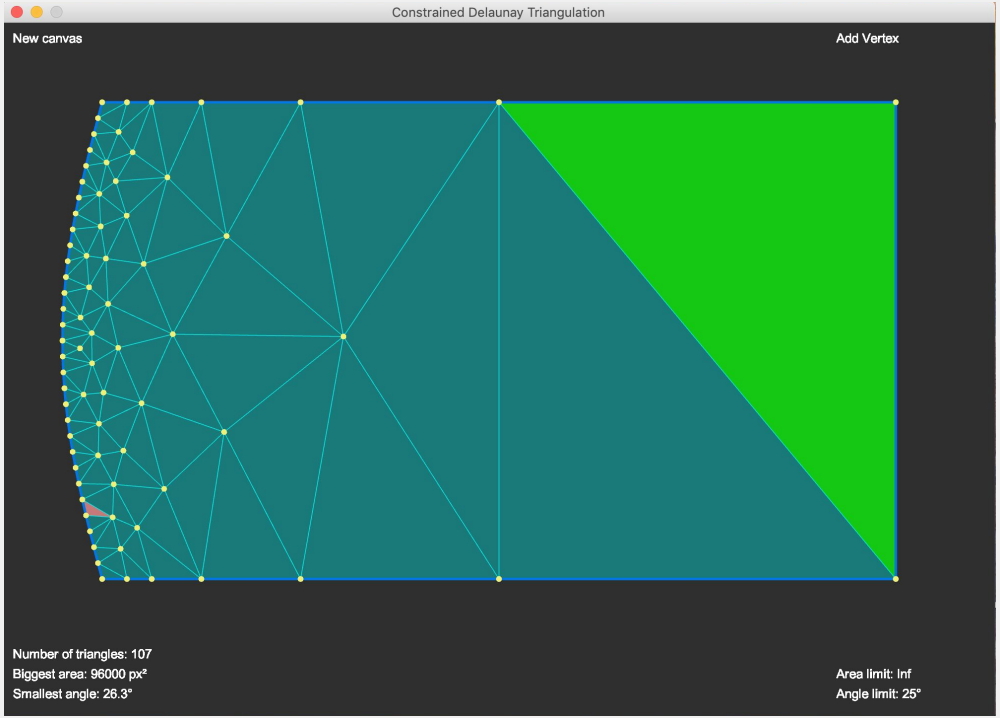


Implementációs adatok

- Kiindulásnak Delaunay háromszögelő lib: https://github.com/jdiemke/delaunay-triangulator
- Java programozási nyelv

- Vizualizálásra optimalizált, nem gyorsaságra

Demó



https://github.com/Dawars/delaunay-triangulator

Források

- http://cg.iit.bme.hu/portal/sites/default/files/oktatott-targyak/3d-geometria-mernoki-visszafejtes/slides/3dgeom-02a-v81.pdf
- https://github.com/jdiemke/delaunay-triangulator
- https://github.com/mattatz/unity-triangulation2D
- https://github.com/peterorlowsky/delaunay-refinement-presentation
- https://github.com/mkacz91/Triangulations