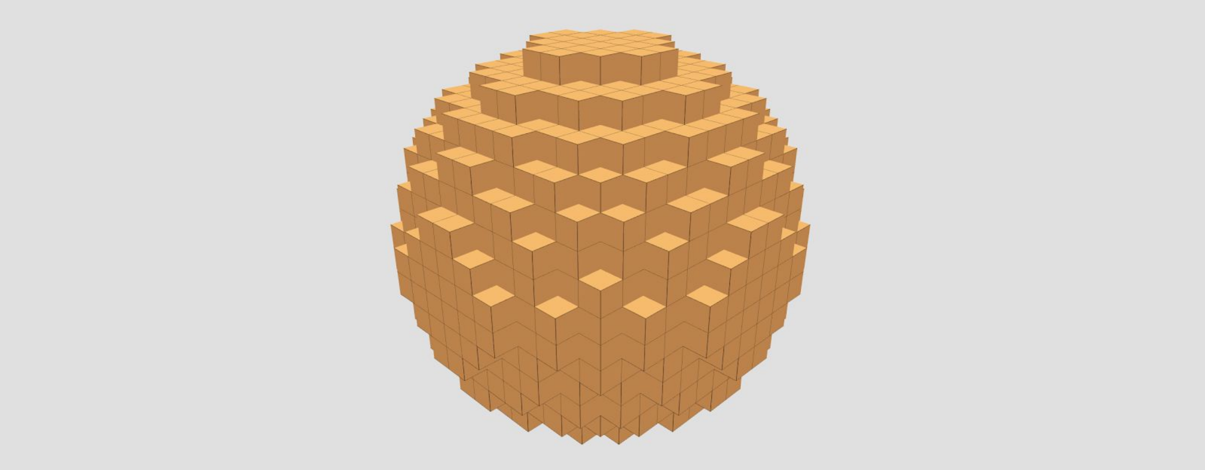
Gravitációs erőtér vizsgálata

(numerikus integrálással)



Feladat:

Numerikus integrálással számítsuk ki egy tömör, egyenletes térfogati tömegsűrűségű gömb gravitációs térerősségét a tér tetszőleges pontjában! A gömböt közelítsük kicsiny kockákkal, amelyek a oldaléle sokkal kisebb a gömb sugaránál ($a \ll R$) (minecraft stílus). Legyen a „minecraftos” gömb teljes tömege M ! Hasonlítsuk össze a gravitációs térerősség irányát és nagyságát az elmélet által jósolt értékekkel, a gömbön kívül és belül. Ábrázoljuk a gravitációs térerősség nagyságát a gömb középpontjától mért távolság függvényében! Hogyan függ az eredmény az a/R aránytól?

Szimulációs információk:

A szimulációkban a geometria teljes tömege mindig megegyezik a Föld tömegével, valamint annak külső átmérője is megegyezik a Föld átmérőjével. Az alább bemutatott példákon szereplő számértékek ezt tükrözik.

Program:

A feladat megoldására készítettem egy JavaScript alkalmazást, és ezt egy weboldal formájában közzé is tettem. Az alább bemutatott példák mind ennek az a felhasználásával készültek.

A weboldal megtekinthető az alábbi linken:

https://kostyalbalint.github.io/Gravity-calculator/

Ehhez hasonlóan a teljes forráskód megtalálhat Github-on az alábbi linken:

https://github.com/KostyalBalint/Gravity-calculator

Az alkalmazás felbontható különböző részegységekre, melyek a:

1. Gravitációs gyorsulás meghatározása a tér egy pontjában
2. A geometria létrehozása
3. 3D megjelenítés
4. A gravitációs gyorsulás megjelenítése

A teljes programkódot nem is mutatnám be, hiszen a nagy része túlmutat a feladat kiíráson. Viszont a feladat megvalósításához fontos 1. és 3. pontra kitérnék.

Gravitációs térerő meghatározása a tér egy pontjában

A kiszámításához szükséges, a felosztott geometria minden elemi pontja, és a mérő pontunk között a

g = G * M / r^2

képletet alkalmazni. Majd az ebből kapott g skalárral megszorozni a mérőpont és a geometria elemi pontja között mutató irányvektort. Majd a geometria minden egyes diszkrét pontjára kapott g vektort összegezni, és ezzel megkapjuk a gravitációs gyorsulást a mérőpontban.

```
gravitys.map((data) => {
  voxelPoints.forEach((voxel) => {
    var r = data.point.distanceToSquared(voxel) * radiusCompensate; //r^2

    var g =  gravityHelper / r;   // g = G * M / r^2

    data.gravity.add(voxel.clone().sub(data.point).normalize().multiplyScalar(g));
  });
  return data;
});
```

Itt a gravitys tömb tartalmazza a mérőpontokat, és a majd hozzá tartozó g vektort is ide tároljuk el. A voxelPoints tömb pedig a geometria pontjait, amelyekhez a gravitációs gyorsulást számítjuk.

Ez a kódrészlet már működő képes, de egy nagyobb geometria felosztás esetén kifejezetten lassú, akár percekig is futhat.

Ám hamar észrevehetjük, hogy ez a számítás párhuzamosítható, mégpedig úgy, hogy minden külön mérőpontra egyszerre végezzük el a műveleteket. Erre tökéletes megoldás a számítógép grafikus processzorát alkalmazni, hiszen az képes akár több ezer műveletet párhuzamosan végezni.

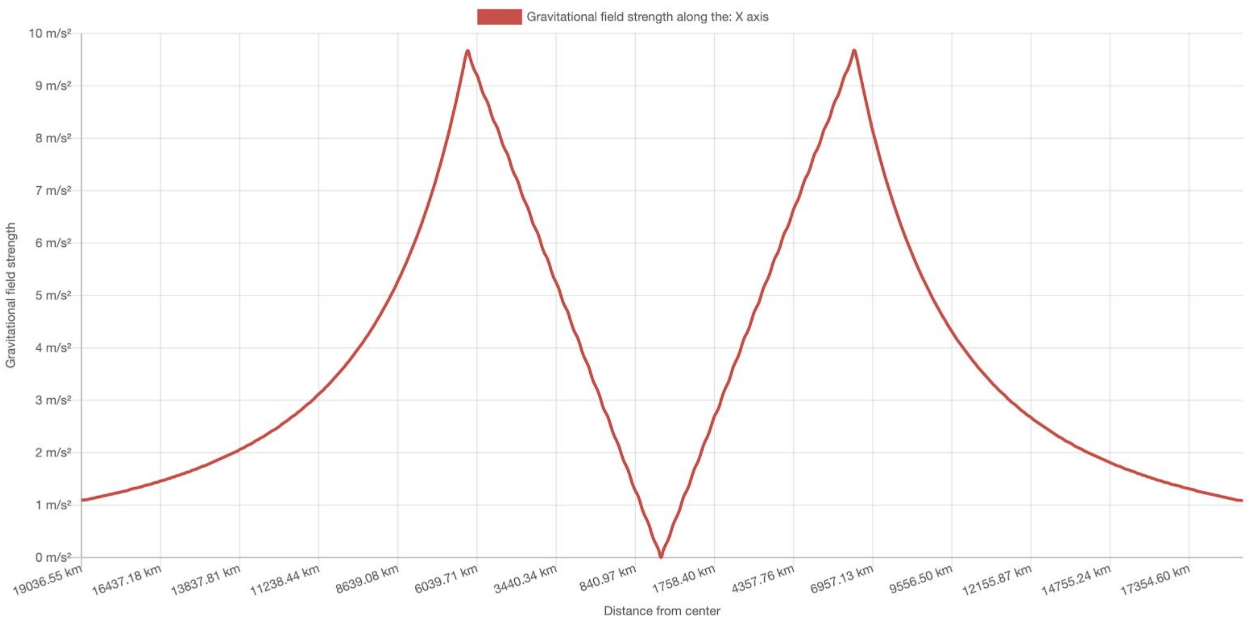
Tehát ezt a megoldást választottam, így még nagyobb felbontás esetén, is néhány másodperc alatt megkapjuk az eredményt.

A gravitációs térerősség megjelenítése

1. Grafikon

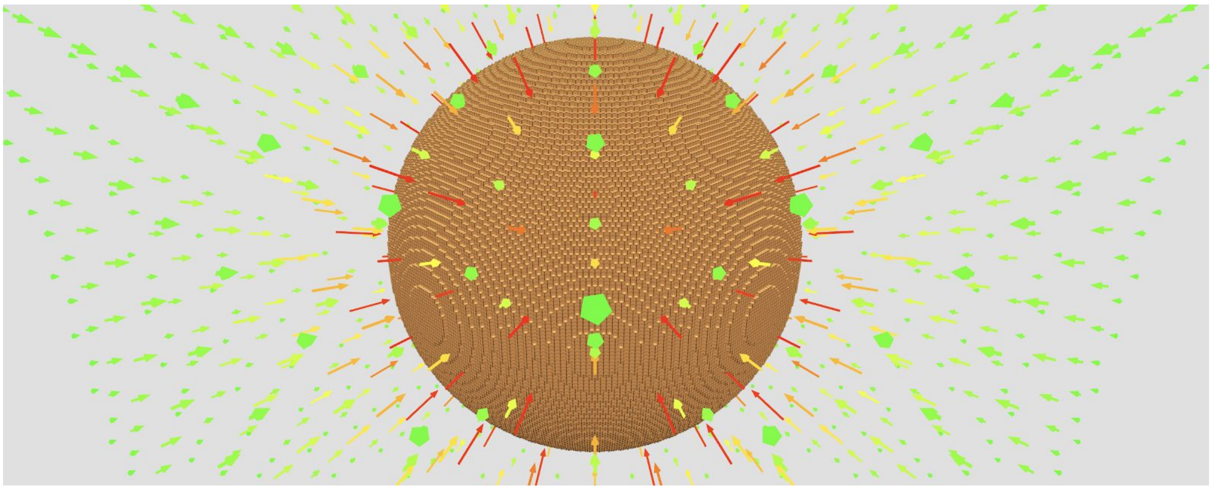
Miután kiszámítottuk a gravitációs gyorsulás vektorokat nincs más dolgunk mint azokat megjeleníteni. Erre két megoldást is elkészítettem.

Az első, hogy a mérőpontokat egy tengely mentén vesszük fel, és a gravitációs gyorsulás vektorok hosszát egy grafikonon jelenítjük meg. Ez egy könnyen érthető módja a gravitációs gyorsulás megjelenítésének.



2. Térbeli vektorok

A megjelenítésnek egy másik módja, hogy a mérőpontokat nem egy egyenes mentén, hanem 3 dimenzióban, egy rács mentén vesszük fel, és a gravitációs gyorsulás vektorokat jelenítjük meg.



Ez a már említett weboldalon megtekinthető, a gömbnél bonyolultabb geometriák esetében is.

A gravitációs gyorsulás irányát a nyíl iránya mutatja, míg ennek a nagyságát a nyíl hossza és színe jelzi. A zöld szín felel meg a leggyengébb, míg a piros a legerősebb gyorsulásnak.

Az alkalmazás használata

Az alkalmazás első betöltésénél a felhasználót az alábbi felület fogadja.

Itt az összes beállítás a Settings fülön tehető meg. Beállítások és magyarázatuk fentről lefelé haladva:

1. Cubic voxel size
   - Az adott geometriát befoglaló kocka felosztás mérete, jelen példában a gömböt befoglaló kocka 64 * 64 * 64 kisebb kockára lesz felosztva
   - A csúszkát mozgatva a felosztás automatikusan frissül, ami mobil eszközök esetén akár pár másodpercet is igénybe vehet.

2. Shape
   - A megjelenített, és szimulált geometria alakja, jelenleg képes, a gömb, a tórusz, és a lapos föld szimulálására.

3. Show texture
   - A geometria színének megjelenítése kapcsolható vele ki, a jobb láthatóság érdekében.

4. Show wireframe
   - A geometria átlátszó drótvázának megjelenítése kapcsolható vele ki, ezzel akár teljesen eltűntetve a geometriát.

5. Show chart points
   - Alapértelmezetten inaktív, de a majdan megjelenítendő grafikon mérő pontjainak a megjelenítését lehet vele kikapcsolni.

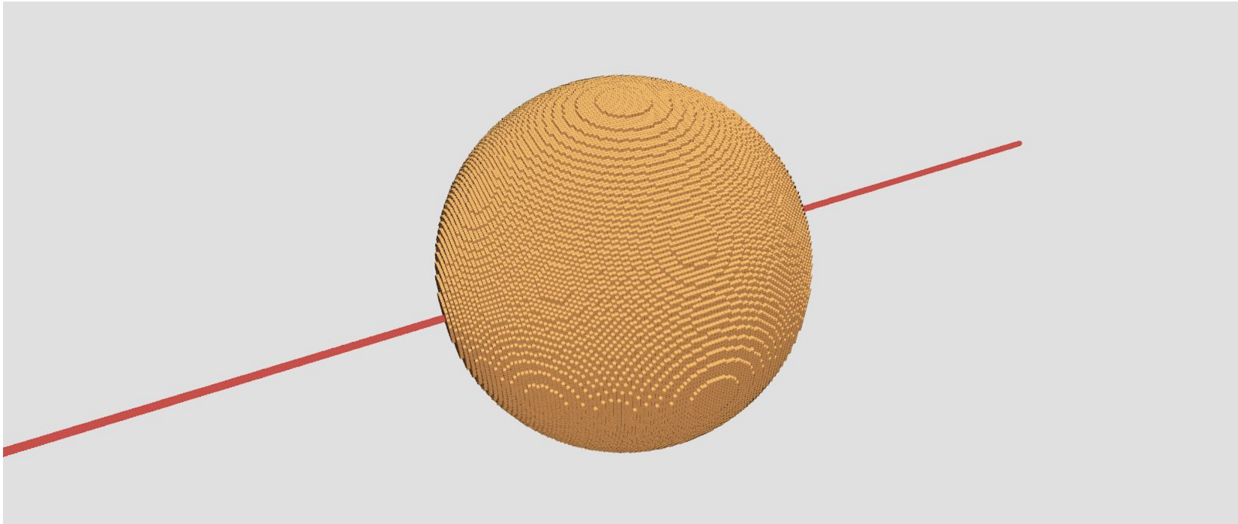
6. Show 3D gravitational field
   - Alapértelmezetten inaktív, de a majdan létrehozott 3 dimenziós gravitációs gyorsulás vektorok megjelenítése kapcsolható ki vele.

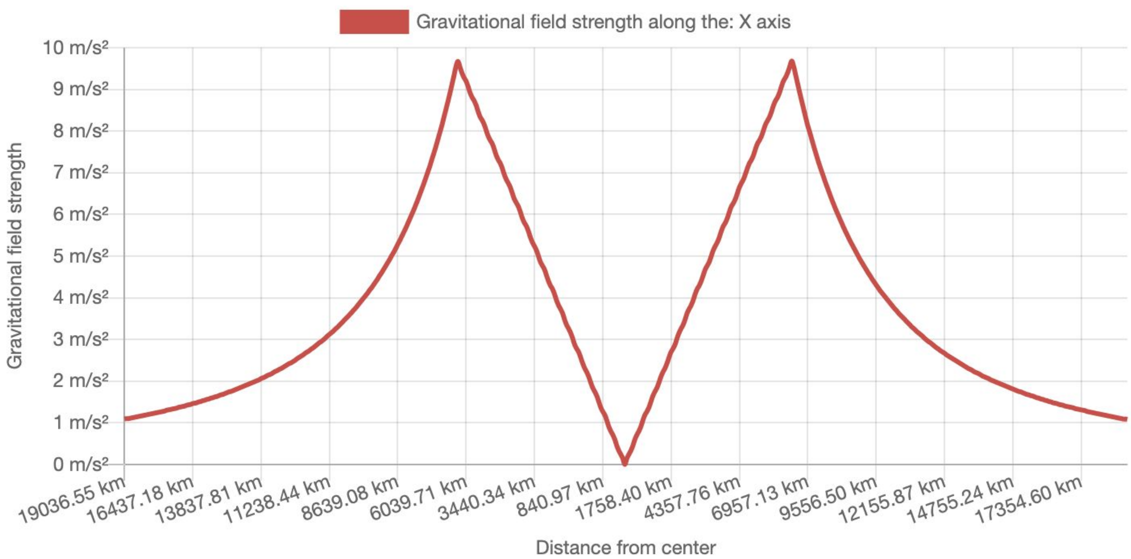
7. Chart generate options
   - A gravitációs gyorsulást megjelenítő grafikon mérő pontjainak iránya állítható be, (melyik tengely mentén történjen a mérés)
   - A grafikonon megjelenő adatok állíthatóak elő a Generate Chart gomb megnyomásával.
   - Ez eltarthat néhány másodpercig, de utána a weboldalon lejjebb görgetve, megtekinthető az interaktív grafikon, ami a középponttól mért távolság függvényében ábrázolja a gravitációs gyorsulás erősségét.

8. Generate 3D gravitational field
   - A 3 dimenzióban megjelenítendő vektorok állíthatóak elő a gomb megnyomásával, amelyek a számítás végeztével meg is jelennek a 3 dimenziós térben a geometria körül.

Szimulációs eredmények

Jó felbontással közelített gömb (128-as felosztás)

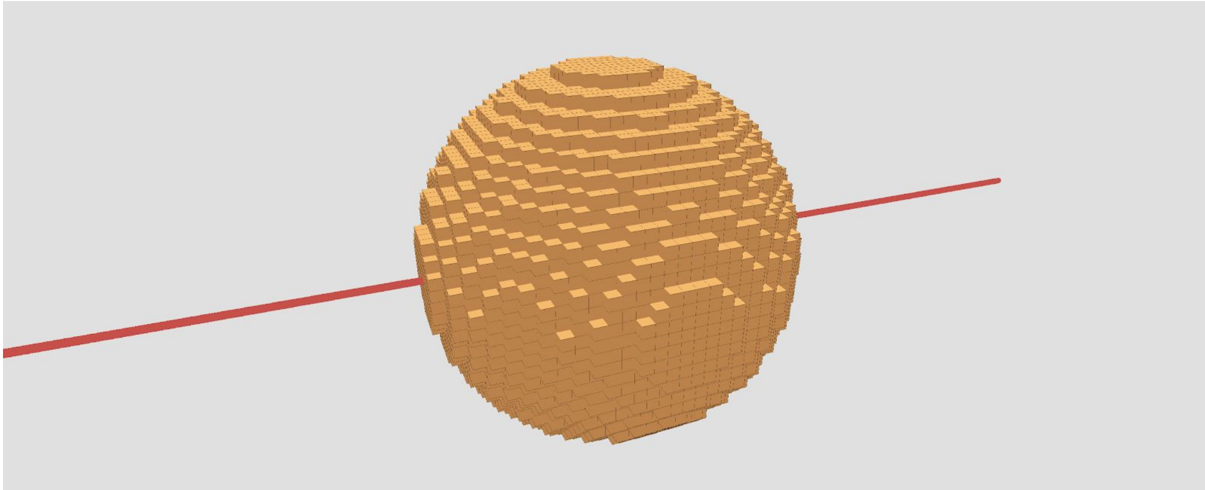


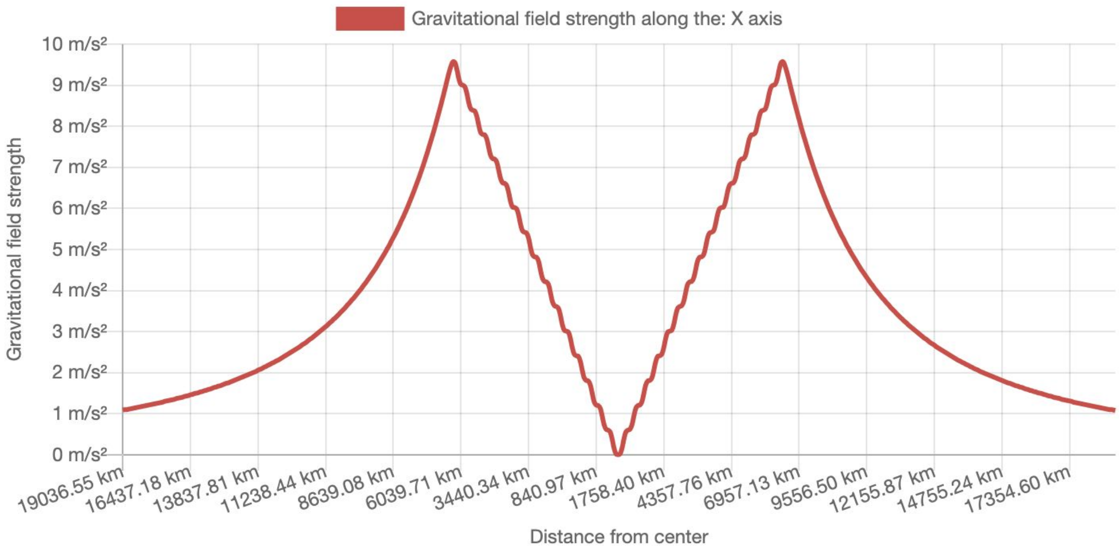


Látszik, hogy a szimuláció tükrözi az elméleti által számított értékeket. A gömb felszínén (6345 km-el a középponttól) a legnagyobb a gravitációs gyorsulás, a szimuláció szerint 9.67 m/s^2 , ami egy egész jó közelítés a földön ismert 9.81 m/s^2 hez.

Az is látszik, hogy amint belépünk a test belsejébe, a gravitációs gyorsulás lineárisan kezd el csökkeni, egészen a 0 m/s^2 -ig a középpontban.

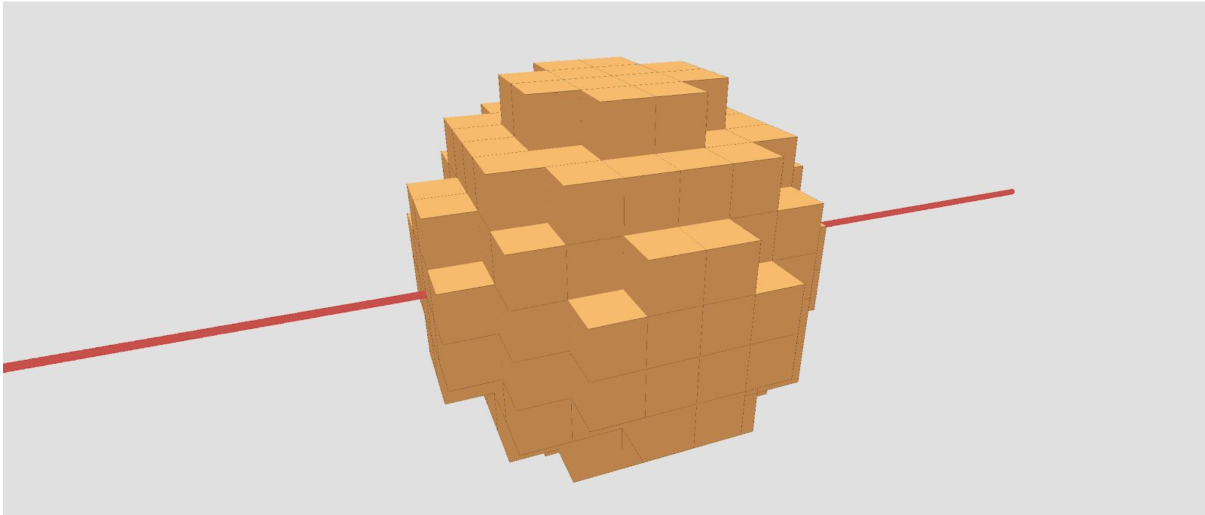
Közepes felbontással közelített gömb (32-es felosztás)

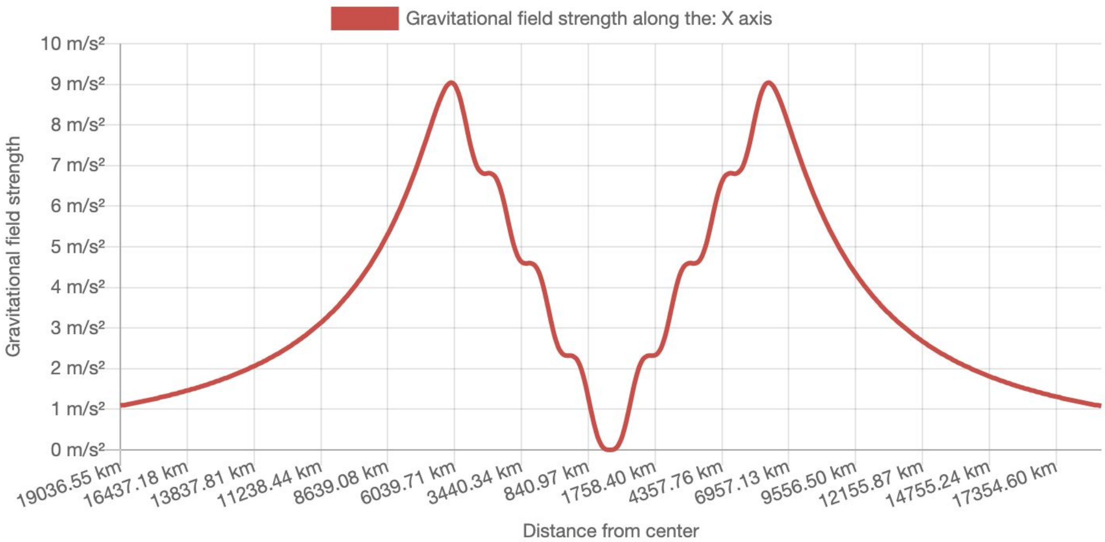




Ennél a felbontásnál már látszik, a kockákkal való közelítés problémája. A testen kívül szinte ugyanazt a görbét láthatjuk, ám a testen belül látszanak a közelítésből adódó lépcsők.

Alacsony felbontással közelített gömb (8-as felosztás)



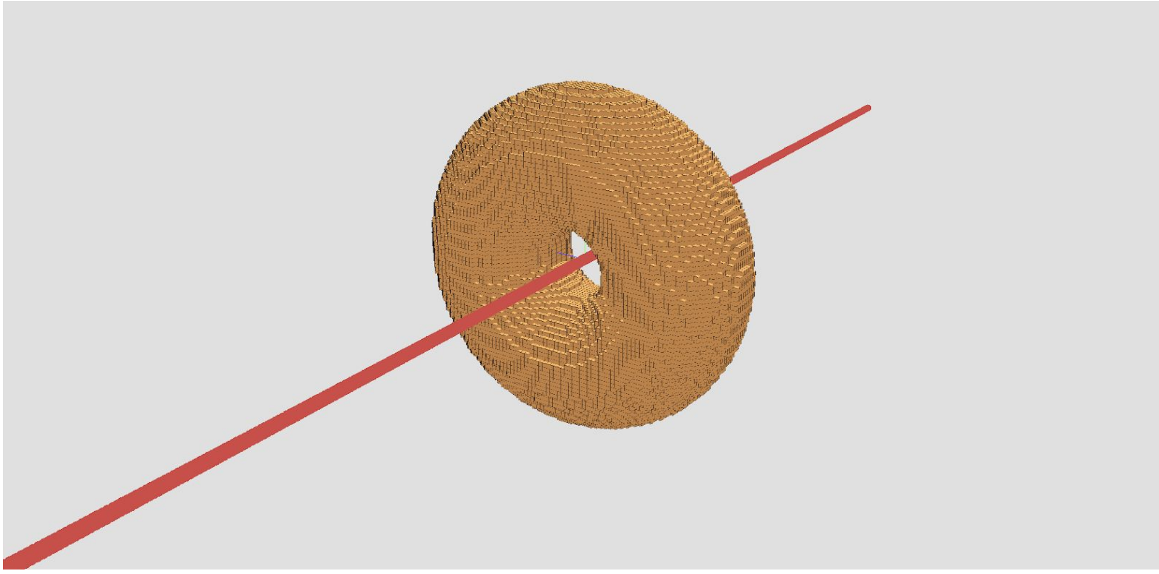


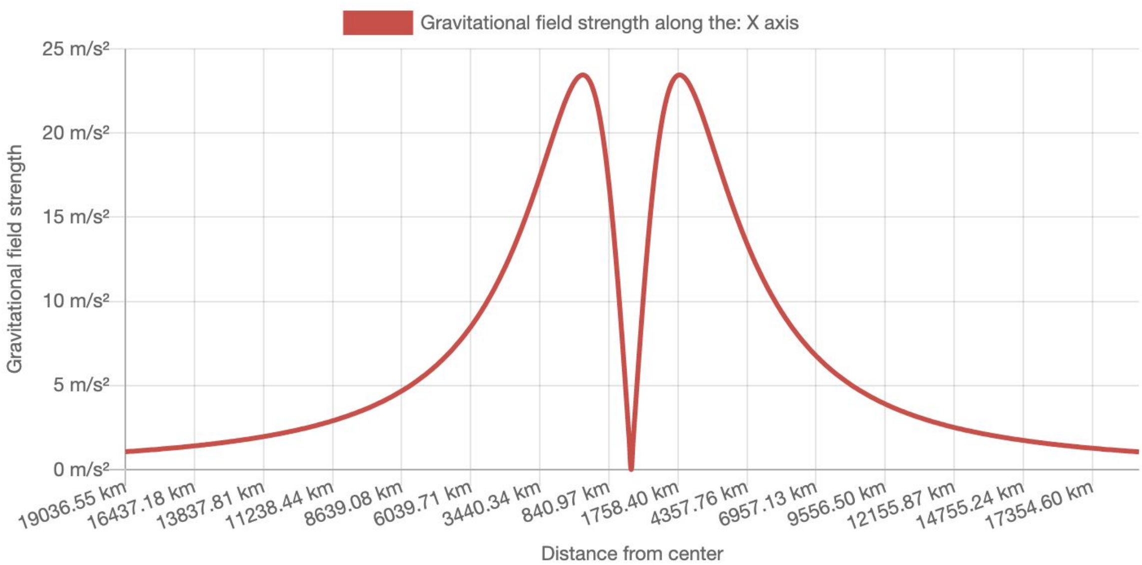
Egy ilyen durva közelítés esetén pedig még szembetűnőbb a már előbb is megfigyelt jelenség.

Érdekesség még, hogy itt már számottevő különbség látszik a felszínen szimulált gravitációs gyorsulás, és a valóság között. Ez talán azért lehetséges, mert jelen közelített gömbnek, nem a sűrűsége, hanem a tömege egyezik meg a földével, és minél durvább a közelítés, annál jobban tér el a szimulált test térfogata a földétől.

Nem gömb alakú testek szimulációja

Tórusz - X tengely mentén

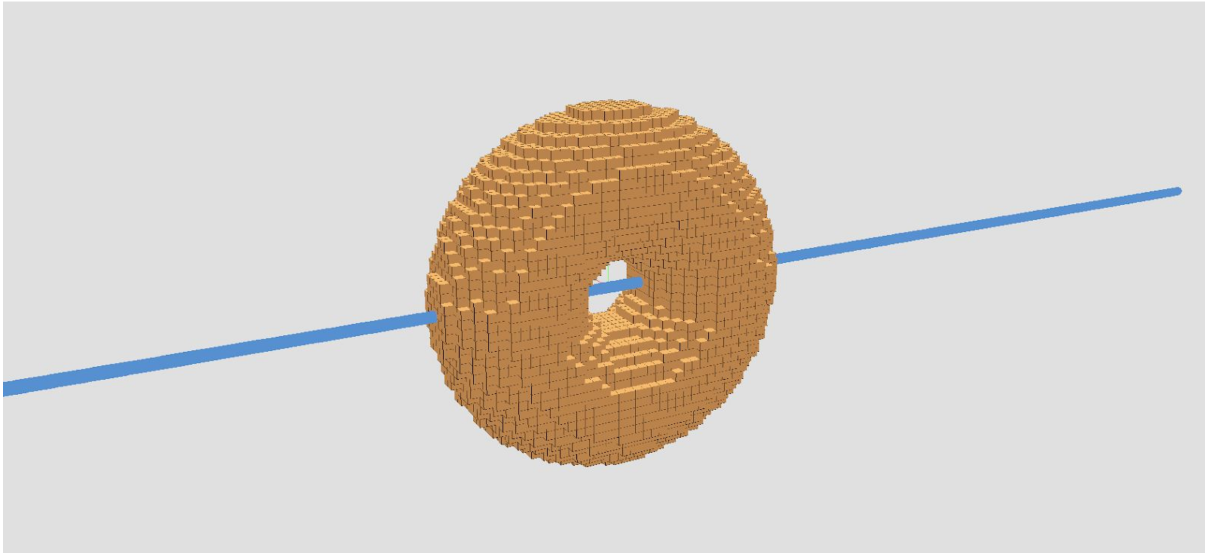


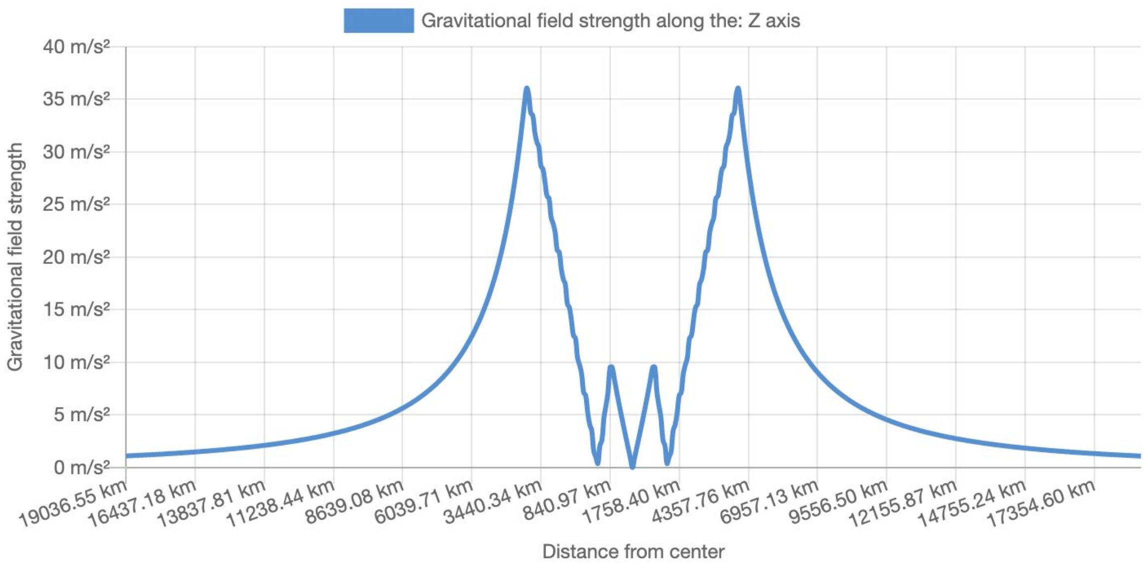


A tórusz tömege megegyezik a föld tömegével, valamit a külső átmérője is azonos a föld átmérőjével.

Érdekes, hogy ebben az esetben a maximális gravitációs gyorsulás 2x-ese a földének. Ez szerintem hasonlóan az előző esetekhez a sűrűséggel függhet össze.

Tórusz - Z tengely mentén





Ez a tórusz beállítás az amire a legjobban kíváncsi voltam, és tudtam, hogy ez a szimuláció miatt olyan programot kell csinálnom, ami nem csak gömbre képes. Érdekes látni ezt a W alakot, ami a gravitációs gyorsulásból adódik, ez logikus, bár elsőre nem biztos, hogy gondol rá az ember.

Tórusz - 3 dimenziós vektorokkal

A végére pedig tényleg csak egy apró érdekesség, ehhez már nem is szeretnék hozzáfűzni semmit.