|  |  |  |
| --- | --- | --- |
|  | **Eötvös Loránd Tudományegyetem**  Informatikai Kar  Térinformatikai és távérzékelési alkalmazások fejlesztése |  |

**Pontfelhő vizualizáció**

Nagy Richárd Tibor, Budapest, 2020

# Tartalomjegyzék

[**Tartalomjegyzék**](#_wahijp2tbo7l)1

[**Bevezetés**](#_vcxiyrhwgzvy)2

[Felhasználói dokumentáció](#_30j0zll) 3

[Rendszerkövetelmények](#_1fob9te) 3

[Telepítés](#_3znysh7) 3

[Indítás](#_2et92p0) 4

[Főmenü](#_quq6rn86asya) 4

[Kamera](#_4d34og8) 4

[Fejlesztői dokumentáció](#_lnxbz9) 4

[Elemzés - pontfelhők megjelenítése Unity-ben](#_35nkun2) 4

[Fejlesztői környezet](#_1ksv4uv) 5

[Felhasználói esetek diagramja](#_44sinio) 6

[Felhasználói esetek leírása](#_2jxsxqh) 6

[A komponensek diagramja](#_zgy635148j0x) 7

[Modell](#_3j2qqm3) 7

[Perzisztencia](#_41mghml) 7

[Nézet](#_2grqrue) 8

[Skálázhatóság](#_2zbgiuw) 9

# 

# Bevezetés

A feladat egy olyan grafikus felületű alkalmazás megvalósítása, amely lehetővé teszi LiDAR (Light Detection and Ranging) pontfelhők interaktív 3 dimenziós vizualizációját. A felületnek támogatnia kell a szokásos megjelenítési funkciókat, úgy mint a navigálás, nagyítás, forgatás, metaadatok tematikus megjelenítése.

# Felhasználói dokumentáció

## Rendszerkövetelmények

| **Operating system** | **Windows** | **macOS** | **Linux** |
| --- | --- | --- | --- |
| **Operating system version** | Windows 7 (SP1+) and Windows 10 | Sierra 10.12+ | Ubuntu 16.04 and Ubuntu 18.04 |
| **CPU** | x86, x64 architecture with SSE2 instruction set support. | x64 architecture with SSE2. | x64 architecture with SSE2 instruction set support. |
| **Graphics API** | DX10, DX11, DX12 capable. | Metal capable Intel and AMD GPUs | OpenGL 3.2+, Vulkan capable. |
| **Additional requirements** | Hardware vendor officially supported drivers. | Apple officially supported drivers. | Gnome desktop environment running on top of X11 windowing system |

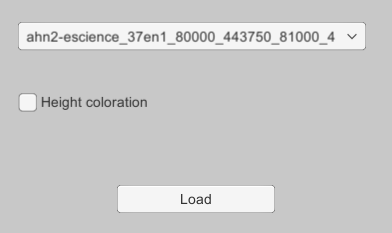
## Telepítés

A program külön telepítést nem igényel.

## Indítás

A projekt betöltésével a Unity környezetbe, vagy lefordított verzió esetén a .exe fájl futtatásával.

## Főmenü



A főmenüben kiválasztható a betöltésre szánt .laz állomány, valamit színezési módot is válaszhatunk.

## Kamera

A kamera támogatja a szokásos megjelenítési funkciókat, a navigálást a WASDQE billentyűk, a forgatást az egér végzi, a kamera gyorsasága pedig a Shift gombbal növelhető.

# Fejlesztői dokumentáció

## Elemzés - pontfelhők megjelenítése Unity-ben

### Minden pont egy Unity GameObject

<https://docs.unity3d.com/ScriptReference/GameObject.html>

A legkevésbé optimális módszer, jelentős valós idejű optimalizációra lenne szükség. (Pl.: távoli pontok elrejtése, nyolcadoló fával)

### Minden pont egy particle a Unity ParticleSystem-ben

<https://docs.unity3d.com/ScriptReference/ParticleSystem.html>

Az alapoktól kezdve nagy mennyiségű elemek megjelenítésére lett optimalizálva, azonban az egyes ParticleEmitter-ek limitáltak.

### Unity DOTs

<https://unity.com/dots>

OOP helyet Data Oriented Programming. Pl.: egy pont objektumokból álló lista helyett tároljunk 3 tömböt, az x, y, és z koordinátákkal. A memória olvasási overhead így drasztikusan csökkenthető.

### Mesh generálás

A beolvasott pontfelhőkből egy mesh-t generálunk. Ha lehetséges, ezzel a módszerrel lenne a leglassabb a betöltés, viszont maga a megjelenítés így lenne a legkisebb költségű.

### Választott módszer

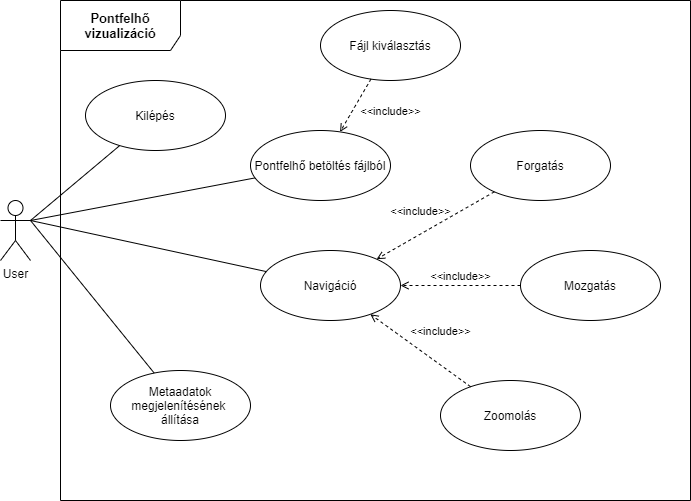
A legjobb módszernek a Unity DOTs és a Unity ParticleSystem ötvözete bizonyult.

## Fejlesztői környezet

A szoftver fejlesztése során az alábbi programokat használtam fel:

* Unity 2019.3
* Visual Studio 2019
* GitKraken
* GitHub

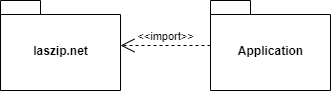
## Felhasználói esetek diagramja



## Felhasználói esetek leírása

|  |  |  |
| --- | --- | --- |
| **Felhasználói eset** | **Leírás** | |
| Indítás | Given: | A fájlkezelőben van | |
| When: | Rákattint a futtatható állományra | |
| Then: | Elindul a program | |
| Kilépés | Given: | Fut a program | |
| When: | Rákattint a kilépés gombra | |
| Then: | A programleáll | |
| Betöltés | Given: | A főmenüben van | |
| When: | Rákattint a betöltés gombra | |
| Then: | Megjelenik a fájl választó menü | |
| Betöltés | Given: | A fájl választó menüben van | |
| When: | Rákattint egy megfelelő fájlra | |
| Then: | A pontfelhő betöltődik | |
| Navigáció | Given: | A pontfelhő betöltődött | |
| When: | Használja a navigációs inputokat | |
| Then: | A kamera elmozdul | |
| Metaadatok megjelenítése | Given: | A pontfelhő betöltődött | |
| When: | Változtat a metaadat opciókon | |
| Then: | A megjelenő metaadatok megváltoznak | |

## A komponensek diagramja

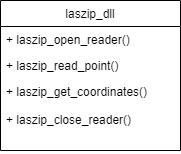


## Modell

A modell a Unity DOTs keretrendszerre épül. A pontok entitásként (Entity) vannak számon tartva, amik a következő komponensekkel rendelkeznek: PointPosition és PointColor.

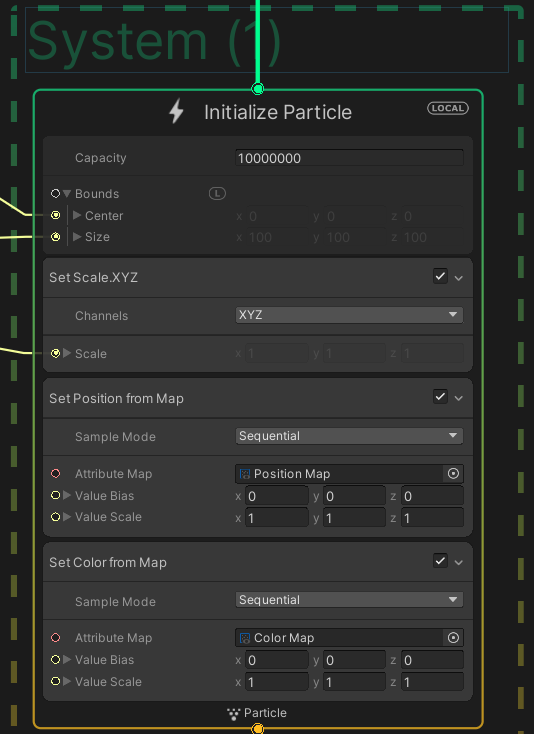
## Perzisztencia

A bináris LAS fájlok beolvasását az integrált laszip.net könyvtár végzi. (<https://github.com/shintadono/laszip.net>)



## Nézet

A megjelenítést a videokártya végzi, a Unity ParticleSystem rendszer felhasználásával. Az egyek pontok shader grafikonja a következő:



## 

## Skálázhatóság

Méréseim alapján egy 1 GB memóriával rendelkező videokártyán egyszerre 10 millió pont jeleníthető meg stabilan. Volt lehetőségem egy 2 GB-os videókártyán is tesztelni a programot, ott a 20 millió pont sem okozott akadályt. Emiatt úgy vélem a program lineárisan skálázható.