# VR Mobil Utveckling 25p

# RAPPORT

# Cavescape

Ludvig Bylund

# Sammanfattning

Detta projekt består utav ett Oculus Quest spel som går ut på att gå ned i en mörk grotta och gräva fram ädelstenar som sedan ska transporteras en bit och läggas i en kista. När man har samlat ihop tillräckligt många ädelstenar så kan man gå till utgången och klara ut spelet. Sedan använde jag olika optimeringstekniker för att få bättre prestanda när spelet eventuellt ska köras på en Oculus Quest.

# Innehållsförteckning

1. Inledning
2. Frågeställningar
3. Metoder
4. Resultat
5. Slutsats

# 1. Inledning

Syftet med denna rapport är att testa olika optimeringstekniker för att få bättre prestanda på ett spel för Oculus Quest. Mitt mål med detta projekt var att se om jag kunde förbättra mitt spels prestanda med några utav de optimeringstekniker jag har/kommer att lära mig under kursens gång. De optimeringstekniker jag har använt är bland annat: Att “baka” ljus, “Occlusion Culling”, och kodoptimering.

# 2. Frågeställningar

Mitt mål med detta projekt var att se om jag kunde förbättra mitt spels prestanda med några utav de optimeringstekniker jag har/kommer att lära mig under kursens gång. Bland annat ville jag testa hur mycket bättre prestandan blir när man “bakar” ljus och använder “Occlusion Culling”, men även om jag kunde använda andra optimeringstekniker.

# 3. Metoder

Baka ljus

Den första metoden jag använde var att “baka” ljus. Min scen har ett antal olika ljuskällor i form av lyktorna och facklan så jag antog att det kan påverka prestandan. Att baka ljus betyder att ljuset på varje objekt kalkyleras och sparas på speciella texturer. Dessa texturer kommer att laddas in när spelet körs så att VR headsetet slipper att beräkna ljuset medans man spelar. Det är lätt att baka ljus, det enda som behövs är att sätta på “static” på ljus källorna som är stilla och alla stilla objekt som kommer att få ljus på sig. Sedan trycker man på “bake” knappen, väntar en ganska lång stund, och sedan är det klart.

## Occlusion Culling

Den andra metoden jag testade var “Occlusion Culling”. Detta är en teknik där alla objekt i spelet som spelaren inte har framför sig, inte renderas. Denna teknik kan vara mycket effektiv på större spel med många, mycket detaljerade objekt. Att sätta på Occlusion Culling är lätt då man endast behöver se till att alla stilla objekt i scenen har “static” på, och sedan kan man trycka på “bake” knappen under “Occlusion” rutan.

## Profilanalys

Den tredje metoden jag använde mig av var att göra en Profilanalys när jag körde spelet för att se om jag kunde hitta exakt vad som tog mest på prestandan. Profilanalyseraren mäter hur många millisekunder det tar att göra varje separat beräkning i bakgrunden medans man spelar.

## Kodoptimering

Den fjärde metoden jag använde var att optimera mina koder. Jag tittade igenom mina koder efter stora problem men det enda jag hittade var det här:

```
// Update is called once per frame
void Update()
{
    if(showButton.action.WasPressedThisFrame())
    {
        menu.SetActive(!menu.activeSelf);

        menu.transform.position = head.position + new Vector3(head.forward.x, 0, head.forward.z).normalized * spawnDistance;
    }

    menu.transform.LookAt(new Vector3(head.position.x, menu.transform.position.y, head.position.z));
    menu.transform.forward *= -1;
}
```

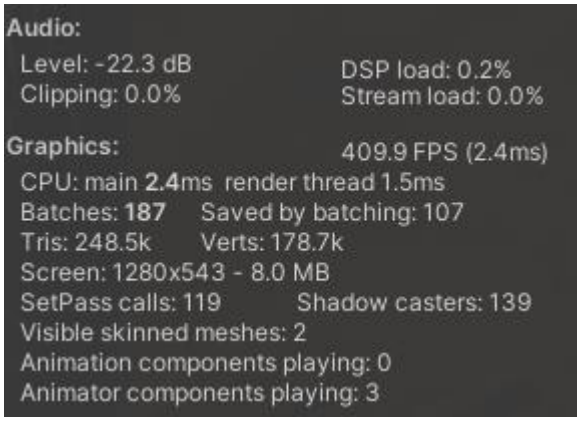
Istället för att ändra menyns transform varje frame så ändras det nu endast när det behövs.

```
// Update is called once per frame
void Update()
{
    if(showButton.action.WasPressedThisFrame())
    {
        menu.SetActive(!menu.activeSelf);
        menu.transform.position = head.position + new Vector3(head.forward.x, 0, head.forward.z).normalized * spawnDistance;
        menu.transform.LookAt(new Vector3(head.position.x, menu.transform.position.y, head.position.z));
        menu.transform.forward *= -1;
    }
}
```

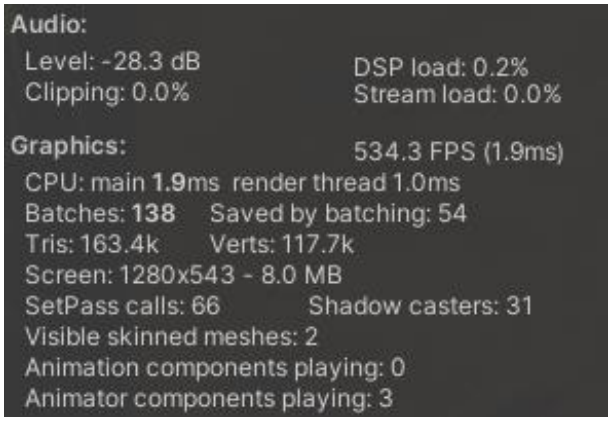
# 4. Resultat

Bakat ljus

Här är spelets prestanda utan bakat ljus:




Här är spelets prestanda med bakat ljus:

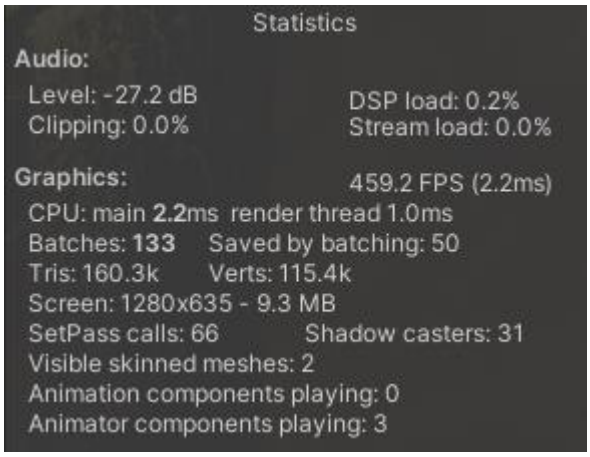



Jag trodde inte att det skulle göra en sådan stor skillnad som det gjorde. En ökning i FPS på ungefär 25% (FPS fluktuerar mycket men det här var bra mittenvärden när jag testade), minskning i Tris på ungefär 35% och Shadow casters minskade markant.

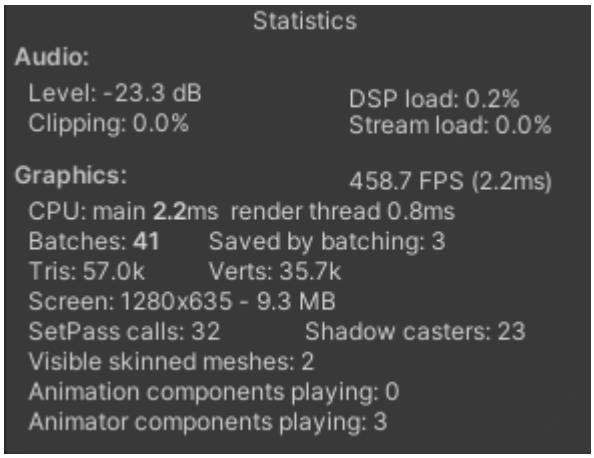
Även om mitt spel är litet med endast 6 ljuskällor som bakades, så gjorde det en märkbar skillnad.

Occlusion Culling

Här är spelet utan Occlusion Culling:



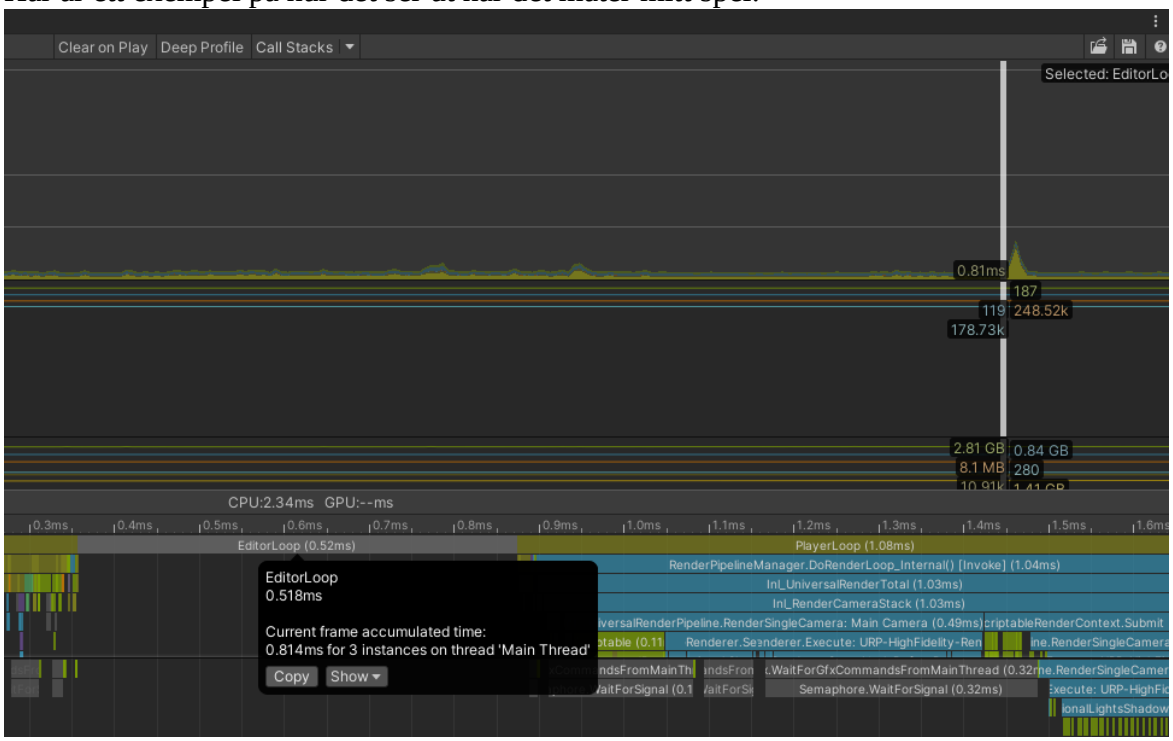

Och här är spelet med Occlusion Culling:



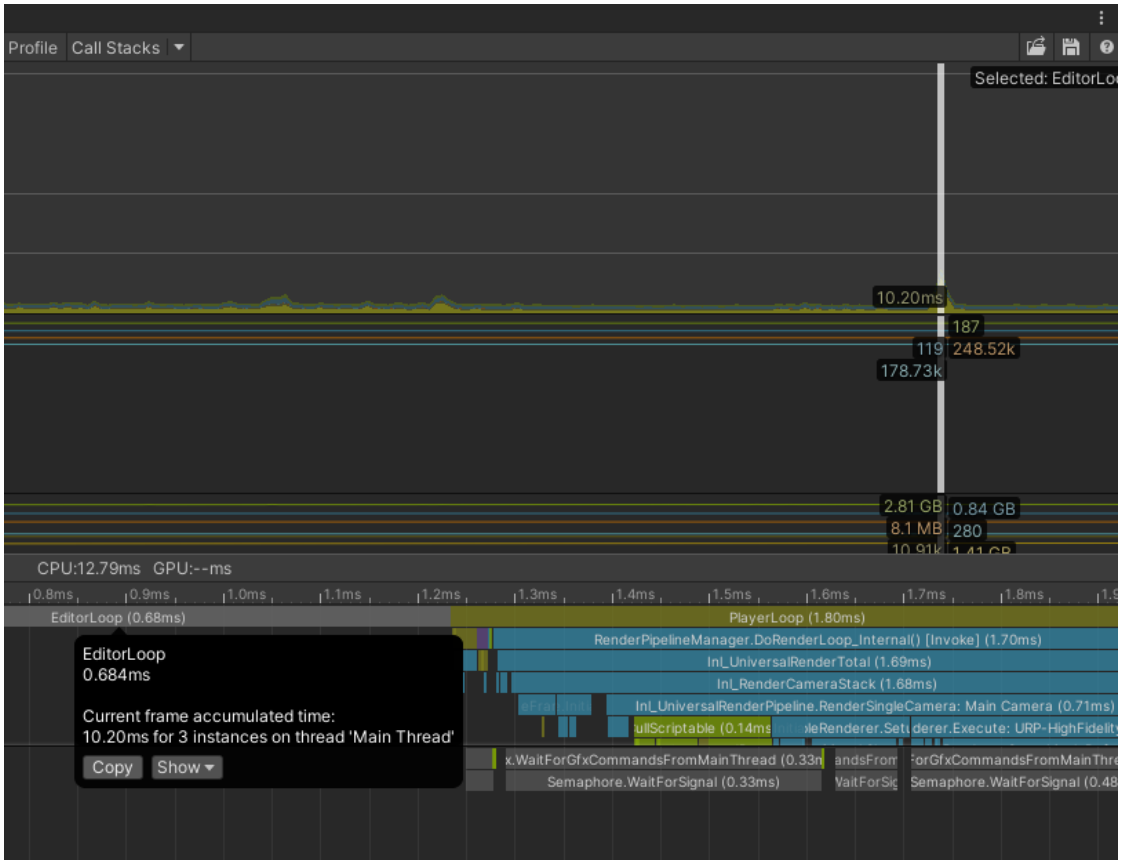

Spelets FPS ändrades inte så mycket, även om mängden “batches” minskade med ungefär 70% och antalet trianglar med ungefär 65%. Det verkar som att spelet är för litet för att effektivt använda Occlusion Culling.

Profilanalys

Här är ett exempel på hur det ser ut när det mäter mitt spel:




Genom några test såg jag att det blev stora spikar med jämna mellanrum. Här är ett exempel på en spik:



Denna spik uppstår på grund av EditorLoop (syns nere till vänster i bilden). EditorLoop är endast ett problem när man testar spelet i Unity, men inte när man har byggt och kör spelet som en separat fil. Så det är inget som behövs optimera.

Annars såg jag inget i profilanalysen som stack ut och drog extra mycket.

Kodoptimering

Jag kunde inte mäta någon märkbar skillnad efter ändringen men på större projekt med många olika scripts kan såna förbättringar sammantaget förbättra spelets prestanda.

## 5. Slutsats

Nytt för mig var optimeringstekniken baka ljus, som jag lärde mig på en av lektionerna. Även på mitt spel med få ljuskällor visade det en mycket bra förbättring i prestanda. Jag kan tänka mig hur effektivt det är på större spel och jag är glad att jag lärde mig detta.

Occlusion Culling, som jag lärde mig från en Youtube video [1], gav inte så mycket effekt på mitt relativt lilla spel. Däremot ser jag stor potential med denna teknik när det kommer till större spel, jag tycker ändå att det var ett lärorikt experiment.

Profilanalysen hjälpte mig inte då jag inte hade några stora fel som stack ut. Däremot kan jag se hur användbart det kan vara om man har några stora och svårhittade problem.

Generellt så kan man säga att med hjälp av optimeringsteknikerna jag testade så kan man öka prestandan på ett spel. Eftersom jag hade ett relativt litet spel utan prestandaproblem innan jag implementerade teknikerna, skulle jag inte säga att det är värt att implementera. Den förbättring som erhålles är inte värt tiden det tar att implementera.

## Referenser:

[1]: Unity Tutorial - Occlusion Culling (in 2 steps)