

Média- és Oktatásinformatikai Tanszék

Naprendszer Szimuláció Exobolygókkal

Dr. Horváth GyőzőTanszékvezető egyetemi docens (ELTE IK)Nyírő Levente Gyula

Programtervező informatikus MSc

Budapest, 2025

Tartalom

[1. Bevezetés 3](#_Toc192858365)

# Bevezetés

Kétéves programtervező informatikus mesterképzésem keretében az utolsó félévben készítenem kell egy diplomamunkát, ahol bizonyíthatom, hogy képes vagyok az adott szakterület fejlesztésére, illetve kutatására. A témám keretein belül kutatási anyagokat kerestem, azokat dolgoztam fel, majd használtam az így megszerzett tudást, hogy a saját programomba beleépítsem, fejlesszem és teszteljem azt. A dokumentációban szóba fog kerülni a témaválasztás, a probléma, ami után kutatni kell és ennek magyarázata, eredményeket fogok ismertetni, beszélni fogok mások eredményeiről, akár más programnyelvekben szerzett tapasztalataikról, majd ismertetni fogom a technikai hátterét az általam írt programnak és az összeszedett megoldásomat a problémára.

Témakörként a Naprendszer modelleket választottam, ugyanis ennek ábrázolása nemcsak lenyűgöző látványvilágot kínál, hanem lehetőséget ad arra is, hogy a modern webes technológiák határait feszegetve, interaktív és valósághű modelleket hozzak létre. Már két féléve foglalkoztam ezzel a témával, hála az egyetemen elérhető kurzusnak, ahol a kiválasztott projektmunkával foglalkozhat a hallgató a félév során és egy mentor is segít a webes tudás elmélyítésében.

A szoftverfejlesztési pályafutásom során mindig is vonzott a frontend fejlesztés, hiszen itt kerül sor a felhasználói felület tervezésére és megvalósítására, amelynek kiemelt szerepe van a felhasználói élmény alakításában. Ehhez kapcsolódóan szerettem volna foglalkozni a háromdimenziós objektumok ábrázolásával weboldalaknál vagy mobilapplikációknál. Ezt azért tartottam fontosnak, mert a webdesign olyan irányba halad, ahol a vizuális megjelenítés és a felhasználóbarát megoldások rendkívül fontosak. A háromdimenziós modellek használata lehetővé teszi, hogy a felhasználók interaktív módon fedezzék fel a tartalmat, ami különösen fontos lehet olyan területeken, mint a tudományos vizualizáció, játékfejlesztés vagy akár az oktatás.

Technikai szempontból a webapplikáció, amiben a kutatásom készült Angular keretrendszerben készült és ebbe lett integrálva a ThreeJS egy canvas-on keresztül. A webes háromdimenziós képi megjelenítésre sok lehetőség elérhető webapplikációknál, ezek közül a legismertebb a ThreeJS, így ez volt az a technológia, amire a kutatásaimat és az általam megírt programot támasztom. Adatlekéréseket is alkalmazok a megírt programban, mégpedig egy francia API-tól kérem el a bolygóknak a pontos adatait.

A ThreeJS már több, mint egy évtizede velünk van az informatikában. 2010-ben publikálták és azóta rengeteg fejlődésen ment keresztül. Ez a JavaScript könyvtár lehetővé teszi, hogy a böngészőben közvetlenül rendereljünk 3D modelleket, anélkül, hogy külső szoftverekre vagy pluginokra lenne szükség. Közben megjelentek az ismert keretrendszerek is a webtechnológiában, mint a React, vagy az Angular, így elő kellett segíteni, hogy ezen technológiákba is importálható legyen a háromdimenziós csomag.

A diplomamunkám végső célja, hogy kutatást végezzek arról, hogyan lehet hatékonyabbá és optimalizáltabbá tenni a ThreeJS működését Angular keretrendszerben, akár mesterséges intelligencia beépítésével a fent említett Naprendszer modellben. Mindez, azért szükséges, mert a megfigyeléseim szerint az egész ThreeJS hatalmas potenciállal rendelkezhet a jövőben, azonban a komplex modellek és a dinamikus animációk kezelése gyakran hatalmas teljesítményigényeket követelhetnek. Ez különösen igaz lehet olyan esetekben, amikor a felhasználói élmény folyamatosan változik és interaktív háromdimenziós tartalmat igényel, mint például a Naprendszer modellek esetében. A kutatásom során olyan technikai megoldásokat és optimalizációs stratégiákat keresek, amelyekkel javítható a renderelési teljesítmény, csökkenthető a memóriahasználat, és biztosítható a felhasználók számára a zökkenőmentes és élvezetes élmény.

# Irodalmi áttekintés

A ThreeJS és a háromdimenziós ábrázolás optimalizálása nem egy új téma a szakmába. Számos kutató és fejlesztő is foglalkozott már ezzel a kérdéssel az elmúlt években. Háromdimenziós tartalmak webböngészőben történő megjelenítése, különösen komplex modellek és dinamikus jelenetek esetében mindig is kihívást jelentett, hogy a teljesítmény és a felhasználói élmény hasonló legyen, mint egy egyszerűbb modellnél. A ThreeJS, mint a WebGL egyik legnépszerűbb JavaScript könyvtára, lehetővé teszi a háromdimenziós modellek hatékony megjelenítését, de a nagy teljesítményigény és a böngésző korlátai miatt a fejlesztőknek gyakran kreatív megoldásokat kell keresniük a problémákra.

Számos kutatás és cikk foglalkozott már a ThreeJS teljesítményének a javításával. Volt, ahol a geometriák egyszerűsítésével és textúrák optimalizálásával foglalkoztak, mindezzel is csökkenthető volt a renderelési idő és a memóriahasználat. Egy másik gyakori megközelítés az instancing. Ez lehetővé teszi, hogy több objektumot egyetlen renderelési hívással jelenítsünk meg, ezzel jelentősen csökkentve a CPU és GPU terhelését.

A modern webes keretrendszerek (Angular, React) is komoly figyelmet kapott az elmúlt pár évben. A komponensalapú technológia és a state management hatékony használata, hogy a háromdimenziós tartalom zökkenőmentesen fusson együtt a webalkalmazás többi részével.

Több kutatás kitér arra is, hogy miképp lehet a mesterséges intelligencia igénybevételével hozzásegíteni a háromdimenziós ábrázolás teljesítményének a fokozását.

Ebben a fejezetben most egyes kutatások eredményeit fogjuk áttekinteni, amik hatással voltak az általam bemutatott kutatásokra. Azt is be fogom mutatni, hogy miképp volt mindez hatással.

## Teljesítményproblémák és optimalizálás JavaScriptben

Az első bemutatott cikk egy 2016-ban megírt publikáció, amely még nem kapcsolódik a ThreeJS-hez. A háromdimenziós megjelenítés weben a sok teljesítményigénytől fogva optimalizálni szükségeltetik. Ehhez meg kell alapozni azt, hogy magának a JavaScript is (amiben fejlesztünk) feltárjuk a problémáit teljesítmény tekintetében és azt vizsgálhassuk, hogy ezt hogy tudjuk optimalizálni. Ezt a tudást átültethetjük a ThreeJS síkjára is és így kaphatunk egy kielégítő teljesítménnyel futó háromdimenziós megjelenítést.

A tanulmány célja, hogy empirikus módszerekkel elemezze a gyakori teljesítményproblémákat és bemutassa, hogyan lehet ezeket optimalizálni. A kutatás során 98 rögzített teljesítményproblémát elemeztek 16 népszerű JavaScript projektből, mind kliensoldali, mind szerveroldali kódokból. A kutatás öt kérdésre keresett választ.

Az első a JavaScript teljesítményromlásának fő okaira próbál magyarázatot találni. Ezeket elsősorban a nem megfelelő API hívásokban, felesleges adatmásolásokban és ismétlődő műveletekben találta meg. Előbbi az esetek 52%-át tette ki a kutatásban. Ez egy elég nagy arány, hogy erre mindenképp odafigyeljen a fejlesztő.

Második a komplexitását kutatta az optimalizációs változtatásoknak. Itt a cikk leírta, hogy a legtöbb optimalizáció csak pár sor kódot érint, mégis ezek vonhatók felelősségre a JavaScript teljesítményromlásáért.

Harmadik esetben az optimalizálás hatására elért teljesítménynövekedést vizsgálja a cikk. Az eredmény jelentős növekedést mutatott ki (átlagosan 20-75%-os növekedés), azonban az eredmények nem voltak konzisztensek különböző JavaScript motorok között.

Negyedik esetben az előző válaszban található optimalizációknak biztosított konzisztens növekedés mértékét vizsgálták, itt azonban mindössze csak 42,68%-a volt minden motorban.

Az utolsó kérdésben a kutatás az ismétlődő mintázatokra tért ki és arra jutott, hogy az összegyűjtött 29 ismétlődő mintázat más projektben is alkalmazhatóak.

## Webes 3D vizualizáció és kihívásai

Ez a cikk az előző cikkhez képest kifejezetten a webes háromdimenziós implementáció során felmerülő teljesítményoptimalizációról szól.

## Háromdimenziós vizualizáció egy bútor webáruházban

Ez a cikk egy olyan online kereskedési platform fejlesztését mutatja be, ahol a cél az volt, hogy a felhasználóknak minél nagyobb élményt adjanak azáltal, hogy a piacra szánt bútorokat pontosan és informatívan mutassa be. Azonban az ilyen fejlesztéseknek a hátránya az is, hogy nagy teljesítmény szükségeltetik és ehhez egy optimalizált frontendet és backendet kell párhuzamosan létrehozni.

A probléma az volt, hogy sok webáruház kétdimenziós képekre támaszkodik, ezzel viszont nem lehet bemutatni pontosan az eladni szánt termék méreteit, dizájnját és funkcionalitását. Ez viszont rosszul meghozott vásárlási döntésekhez, magasabb visszaküldési arányokhoz és elégedetlen ügyfelekhez vezet, különösen olyan iparágakban, mint a bútorkereskedelem, ahol a térbeli vizualizáció kulcsfontosságú.

A webáruház a ThreeJS integrálása mellett határozta el magát, ezzel elérhetővé téve a felhasználónak a termékek 360 fokos forgatását, közelítést a részletek megjelenítéséhez, illetve valós időben testre szabhatják az anyagokat és a színeket.

További megoldásként a MERN stack-re hivatkozik a cikk, ez annyit takar, hogy ezáltal a fejlesztés során létrejövő architektúra skálázhatóvá és modulárissá válik, mind frontend, backend és adatbázis szinten is. Ezzel elérve, hogy a teljesítmény a legjobbá váljon kevesebb erőforrás igénybevételével. Ez is egy fontos szempont, hogy az ügyfélelégedettség növekedésnek induljon.

A kutatás kimutatta az elért előnyöket is, legyen szó a fokozott felhasználói elköteleződéstől, vagy épp a magabiztosabb vásárlói döntéshozataltól. Jelentősen csökkent továbbá a visszáruk száma is, ugyanis a pontos vizualizáció csökkentette az eltéréseket a várakozások és a valóság között.

## Fényhasználat optimalizálása háromdimenziós megjelenítésben

Ebben a cikkben a fényhasználat optimalizálását vizsgálják. Ez az én kutatásomban szintén egy fontos támaszpont, ugyanis a fénybeállítások általában több erőforrást tudnak igénybe venni és ezáltal nagyobb optimalizációra van szükségünk, hogy egy élvezhető teljesítményélményt lehessen adni a felhasználóknak. A cikkben kitérnek arra is, hogy a megfelelő beállítások a renderelési időt is jelentősen le tudják csökkenteni.

A fényforrások helyes beállítása elengedhetetlen a valósághű árnyékok, tükröződések és fényvisszaverődések létrehozásához. Ennek optimalizálása segít kiemelni a modellek részleteit, egyben mélységérzetet ad és növeli a vizuális hitelességet. Több fénytípust is megkülönböztetünk. Környezeti fény – alapvető megvilágítás, ami az egész teret bevilágítja. Irányított fény – ilyen a napfény is. Pontszerű fény – lámpa fényéhez hasonlítható. Az utolsó ilyen a fókuszfény – ami egy zseblámpa világítását imitálhatja.

Az optimalizálási technikák sokrétűek lehetnek, ezt a kutatás is leírja. Dinamikus árnyékok esetén, ahol a valós idejű árnyékszámítások kellenek a pontosabb megjelenítés céljából erőforrásigényesek lehetnek. Optimalizálás céljából ezt csakis a kritikus területeken érdemes használni. Másik ilyen technika az előre számított fény- és árnyékinformációk. Ezek csökkentik a valós idejű számítások terhelését, ami mindenképp előnyös renderelés szempontjából. Ezen kívül még a fénymintavételezés az, ami segítheti az optimalizálást. Itt a fényforrások hatásának mintavételezése csökkenti a renderelési időt.

Következtetésképpen a fényoptimalizálás, ahogy azt a fenti technikák is leírják, nemcsak a vizuális minőséget javítja, hanem egyben csökkenti is a renderelési időt és a hardverigényt. Arra is érdemes a fejlesztés során figyelni, hogy a felesleges fényforrásokat a fejlesztő eltávolítsa és a fényterheket jelentősen csökkentse, ugyanis ez egyben teljesítménynövekedést is eredményezhet. Ezekre a technikákra nagyon odafigyelnek az olyan gyakorlati területeken is, mint a játékfejlesztés, építészeti vizualizáció, vagy épp a termékdesign.