Pannon Egyetem

Műszaki Informatikai Kar

[Tanszék neve]

[Szak neve]

**SZAK-/DIPLOMADOLGOZAT**

**[Dolgozat címe]**

**[Név]**

Témavezető: [Témavezető neve]

Külső/belső konzulens: [Konzulens neve]

[évszám]

Témakiírás

A szkennelt formában megkapott témakiírás beillesztése a dolgozatba.



Hallgatói nyilatkozat

Alulírott <<név>> hallgató (Neptun kód: <<SAJÁT NEPTUN KÓD>>) kijelentem, és a dolgozat feltöltésével egyidejűleg nyilatkozom, hogy a <<dolgozatcím>> című <<záródolgozatot/szakdolgozatot/diplomadolgozatot>> (a továbbiakban: dolgozat) a Pannon Egyetem Műszaki Informatikai Kar <<szervezeti egységében (tanszékén)>> készítettem a <<végzettség>> oklevél megszerzése érdekében.

Kijelentem, hogy a dolgozatban csak a megadott és hivatkozott forrásokat használtam fel, és ezekre a vonatkozó idézési szabályok szerint hivatkoztam.

Nyilatkozom, hogy a dolgozat érdemi része saját szellemi alkotásom eredménye, és azt más intézményben, szakon, vagy felsőfokú képesítés megszerzésére nem nyújtottam be. Tudomásul veszem, hogy a plágium vagy szerzői jogsértés esetén a dolgozatom elutasításra kerülhet, és ellenem fegyelmi eljárás indulhat. Tudomásul veszem továbbá, hogy szerzői jogsértés esetén az Egyetem jogosult a dolgozat elérhetőségét korlátozni, valamint eltávolítani a dokumentumot a dolgozatok tárolására szolgáló, a témát vezető szervezeti egység által meghatározott elektronikus zárt rendszerből.

Tudomásul veszem továbbá, hogy a Pannon Egyetem a dolgozat eredményeit saját céljaira eltérő írásbeli megállapodás hiányában a Pannon Egyetem Szellemi Tulajdon Kezelési Szabályzatában foglaltaknak megfelelően szabadon felhasználhatja.

Nyilatkozom, hogy a dolgozat elkészítése során mesterséges intelligencia eszközöket *használtam /nem használtam[[1]](#footnote-1).*

Nyilatkozom, hogy a dolgozat elkészítése során az alábbi táblázatban feltüntetett mesterséges intelligencia eszközöket kizárólag a kutatási, illetve fejlesztési feladat támogatására használtam fel, az érdemi munka, elemzés és következtetések teljes mértékben saját szellemi alkotásomat képezik.

Példa a táblázat kitöltésére:

|  |  |  |  |
| --- | --- | --- | --- |
| **Alkalmazott technológia** | **Alkalmazás módja** | **Előállított tartalom** | **MI használat aránya** |
| GPT-4o (OpenAI) | szöveges összefoglaló generálása | 2.2 fejezet | 80% |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Dátum: Veszprém/Nagykanizsa/Zalaegerszeg, 20.. <<hónap neve>> <<nap>>.

*<<hallgató neve>>*

Témavezetői nyilatkozat

Alulírott <<témavezető neve>> témavezető kijelentem, hogy a <<dolgozatcím>> című dolgozatot *<<*hallgató neve*>>* a Pannon Egyetem <<tanszék neve>>én készítette <<végzettség>> végzettség megszerzése érdekében.

Kijelentem, hogy a dolgozat védésre bocsátását engedélyezem.

Dátum: Veszprém/Nagykanizsa/Zalaegerszeg, 20.. <<hónap neve>> <<nap>>.

*<<témavezető neve>>*

Köszönetnyilvánítás

A hallgató köszönetet nyilvánít mindazoknak, akiktől (elméleti, gyakorlati, erkölcsi stb.) segítséget kapott.

Tartalmi összefoglaló

Tartalmi összefoglaló magyarul. Az összefoglalónak tartalmaznia kell (rövid, velős és összefüggő megfogalmazásban) a következőket:

* téma megnevezése,
* megoldott feladat megfogalmazása,
* megoldási mód,
* elért eredmények,
* kulcsszavak (4-6 darab)
* terjedelme nem lehet több 1 A4-es oldalnál.

Az összefoglalót magyar és angol nyelven kell készíteni. Sorrendben a dolgozat nyelvével megegyező kerül előrébb. A cím Title stílusú, formázása: Times New Roman, nagybetű, 14 pt, félkövér, középre igazított; az összefoglaló Normál stílusú, formázása: Times New Roman, 12 pt, sorkizárt, 1.5-ös sortávolság.

**Kulcsszavak:** [4-6 kulcsszó felsorolása, vesszővel elválasztva]

Abstract

Abstract in English

**Keywords:** [list 4-6 keywords]

Tartalomjegyzék

[Jelölésjegyzék 8](#_Toc99107193)

[1. Fejezet 9](#_Toc99107194)

[1.1. Alfejezet 9](#_Toc99107195)

[1.1.1. Alfejezet 9](#_Toc99107196)

[2. Új fejezet 10](#_Toc99107197)

[2.1. Új alfejezet 10](#_Toc99107198)

[3. Instrukciók 11](#_Toc99107199)

[Irodalomjegyzék 12](#_Toc99107200)

[Mellékletek 13](#_Toc99107201)

[Ábrajegyzék 14](#_Toc99107202)

[Táblázatjegyzék 15](#_Toc99107203)

Jelölésjegyzék

|  |  |
| --- | --- |
| AI: | Artificial Intelligence (Mesterséges Intelligencia) |
| GPU: | Graphical Processing Unit (Grafikus Processzor / Grafikus Feldolgozó Egység) |
| API: | Application Programming Interface (Alkalmazásprogramozási Felület) |
| CPU: | Central Processing Unit (Központi Feldolgozó Egység / Processzor) |
| GUI: | Graphical User Interface (Grafikus Felhasználói Felület) |
| HCI: | Human Computer Interaction (Ember-gép kapcsolat) |
| CIS: | Cognitive Information System (Kognitív információs rendszer) |

# A\* útkereső algoritmus és vizualizálása

A projekt keretében egy A\* útkereső algoritmus került implementálásra, melynek működése grafikusan is ábrázolásra került. A megvalósítás Python programozási nyelven, a Raylib grafikus könyvtár (Python-kötéssel) segítségével történt. A Python, mint magas szintű programozási nyelv, hatékony és gyors fejlesztést tett lehetővé.

A grafikus megjelenítéshez a Raylib könyvtár került felhasználásra. A Raylib egy nyílt forráskódú, eredetileg C nyelven írt könyvtár, amely Python-kötéssel is rendelkezik, és alkalmas 2D, valamint 3D-s alkalmazások és játékok fejlesztésére. Előnye, hogy számos beépített funkcióval rendelkezik, ugyanakkor teret enged az egyedi megoldások implementálásának is. Más játékmotoroktól eltérően nem rendelkezik grafikus szerkesztőfelülettel, így a teljes fejlesztési folyamat a kódban zajlik.

A képen képernyőkép, Téglalap, tervezés, Betűtípus látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

*1. ábra: Raylib python verziójának a logója*

## Útkereső algoritmus megvalósítása

Az útvonaltervezési feladathoz az A\* algoritmus került kiválasztásra. Az A\* egy széles körben alkalmazott gráf- és útvonalkereső algoritmus, amelyet többek között a játékfejlesztés, a robotika és a navigációs rendszerek területén is használnak.

Az A\* egy informált keresési algoritmus, amely a legrövidebb, illetve legkisebb költségű útvonal meghatározásához az útvonalköltségeket és egy heurisztikus becslést használ. Az algoritmus minden egyes csomóponthoz (állapothoz) egy költségértéket (súlyt) rendel a következő képlet alapján:

***f(n)= g(n) + h(n)***

ahol:

* **g(n):** A kezdőponttól az aktuális csomópontig megtett út tényleges költsége.
* **h(n):** Az aktuális csomóponttól a célpontig hátralévő út becsült költsége (heurisztika).
* **f(n)**: Az aktuális csomópont teljes becsült költsége.

Az algoritmus minden lépésben azt a csomópontot terjeszti ki, amely a legalacsonyabb f(n) értékkel rendelkezik. A h(n) heurisztikus függvény a megoldandó problémától függ. Geometriai távolság becslésére gyakran használatos az Euklideszi-távolság:

Az algoritmus lépései a következők:

1. A kezdő csomópont felvétele a nyílt listára (a potenciális következő lépések listája).
2. A nyílt listáról a legalacsonyabb ***f(n)*** költségű csomópont kivétele és megvizsgálása.
3. Ha a vizsgált csomópont a cél, az algoritmus véget ér.
4. Ellenkező esetben a csomópont kiterjesztése és a szomszédos csomópontok vizsgálata.
5. Minden szomszédra kiszámításra kerülnek az ***f(n)***, ***g(n)*** és ***h(n)*** értékek.
6. Ha egy már vizsgált szomszédos csomópont egy új útvonalon keresztül alacsonyabb költséggel érhető el, az értékei frissítésre kerülnek.
7. A folyamat a 2. lépéstől ismétlődik, amíg a cél elérésre nem kerül, vagy a nyílt lista kiürül.

Nyílt lista: Potenciális következő lépések listája/sora, aminek első eleme a legolcsóbb mind közül.

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

*2. ábra: Részlet a költségszámítás megvalósításáról Python kódban*

Az A\* algoritmus előnyei:

* Megfelelő heurisztika alkalmazása esetén garantáltan megtalálja a legoptimálisabb útvonalat.
* A heurisztika használatával hatékonyan szűkíti a keresési teret.
* Széles körben alkalmazható.

És hátrányai:

* Jelentős memóriát igényelhet a csomópontok tárolása miatt.
* Nem megfelelő heurisztika esetén a teljesítménye lecsökkenhet.
* Bizonyos problémák esetén a heurisztika definiálása nehézségekbe ütközhet.

Ezeket a pontokat szem előtt tartva került kiválasztásra az A\*. A feladat megoldására két lehetséges megközelítés került feltárásra:

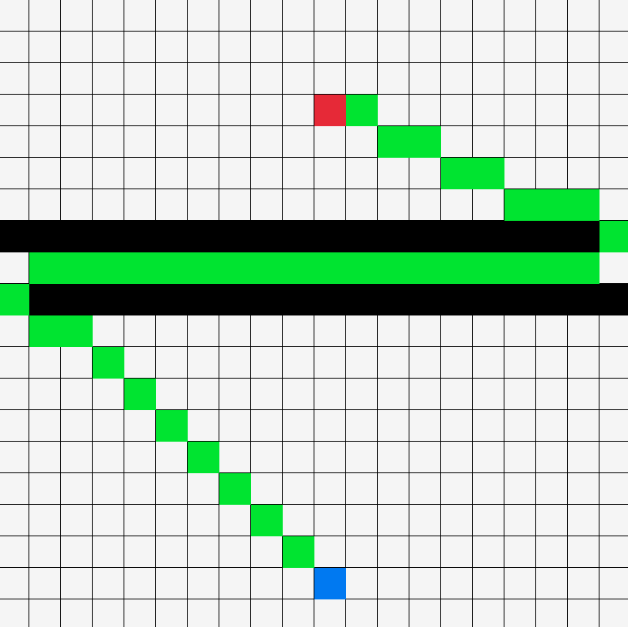
* Gridmap (négyzetháló) alapú megvalósítás.
* Útszegmensek használata csomópontokként.

### Gridmap (négyzetháló)

Az A\* algoritmust gyakran alkalmazzák négyzethálós rendszereken, ahol minden egyes négyzet egy lehetséges állapotot (csomópontot) képvisel. Ez a módszer alkalmazható diszkrét, rács alapú térképeken, vagy egy folytonos térre (pl. úthálózatra) vetített hálón, amelyet "navmesh"-nek is neveznek.

Egy csomópont (négyzet) a következő attribútumokat tárolhatja:

* Az algoritmus működéséhez szükséges ***f***, ***g*** és ***h*** értékek.
* A négyzethálóban elfoglalt sor- és oszlopindex.
* A négyzet x és y koordinátái a grafikus megjelenítéshez.
* Hivatkozás a szülő csomópontra, amelyből az aktuális állapot elérhető volt (az útvonal rekonstrukciójához).
* Speciális paraméterek, mint például a járhatóság vagy a mozgás költségét módosító tényezők (pl. akadályok, nehezebben járható terep).



3. ábra: Egy négyzetrácsos pályán a kék és piros pont közötti legrövidebb út zölddel jelölve, ahol a fekete négyzetek nem átjárhatóak

Bár ez a megoldás egyszerű, kétdimenziós térképek esetén hatékony, kanyargós, valósághű úthálózatok modellezéséhez pontatlan lehet.

A képen tér, Színesség, pixel, sor látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

*4. ábra: Próbálkozás egy kanyargós úttal egy négyzetrácsos pályán*

Lehetséges lenne a négyzethálót a bejárni kívánt útszakaszra vetíteni, majd az útvonal meghatározása után a pontatlanságokat utólagosan korrigálni, azonban ez a megközelítés nem tekinthető hatékonynak, mivel létezik ennél elegánsabb megoldás.

### Utak, út szegmensek használata, mint állapot

Egy alternatív megközelítés a négyzetháló elhagyása, és ehelyett a már meglévő úthálózat csomópontjainak (pl. útkereszteződések, útszakaszok végpontjai) állapotokként való kezelése. Ez a módszer feltételezi egy előre definiált úthálózat meglétét, ahol az összeköttetések és azok koordinátái ismertek.

A képen művészet látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

*5. ábra: Úthálózat felsőnézete, láthatóak az út szegmensek összeillesztései, melyekhez koordináta is tartozik*

Az algoritmus működési elve változatlan marad: a keresés a kezdőponttól indulva, a legkisebb költségű csomópontok (útszegmens-végpontok) kiterjesztésével halad a cél felé. Ebben az esetben a h(n) heurisztika az aktuális csomópont és a cél közötti légvonalbeli távolság alapján számítható. A két módszer összehasonlítása után kerül kiválasztásra a projektben véglegesen használt megoldás.

## Fejlesztési tapasztalatok és megvalósítás

A bemutató program fejlesztése a Raylib könyvtár Python nyelvű verziójának felhasználásával történt. A program célja az volt, hogy zárt, kontrollált környezetben lehessen tesztelni az algoritmus működését, tulajdonságait, skálázhatóságát és általános megvalósítását.

A későbbi fázisokban az algoritmus a Webots szimulációs környezetben kerül alkalmazásra, ahol fizikai modellként működő jármű fogja használni.

A Raylib könyvtár korábbi projektek során már bevált eszköznek bizonyult, így logikus választás volt a tesztprogram fejlesztéséhez. Az A\* algoritmus implementálásával korábban is történtek kísérletek, azonban a mostani megvalósítás stabilabb és funkcionálisabb alapot biztosított a további fejlesztésekhez.

### Program grafikus megvalósítása

A Raylib könyvtár az alábbi kódrészlettel teszi lehetővé egy ablak létrehozását:

A képen szöveg, képernyőkép, Betűtípus látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.

*6. ábra: Kódrészlet egy játékablak létrehozásához*

*A képen képernyőkép, szöveg, diagram, tervezés látható

Előfordulhat, hogy az AI által létrehozott tartalom helytelen.*

*7. ábra: A kódrészlet eredménye*

* init\_window() – Létrehozza az alkalmazás ablakát a megadott paraméterek alapján (szélesség, magasság, ablak címe).
* while not window\_should\_close() – A program fő ciklusát valósítja meg; a ciklus mindaddig fut, amíg a felhasználó nem zárja be az ablakot.
* begin\_drawing() / end\_drawing() – A megjelenítéshez szükséges függvénypár; minden, ami a két hívás között szerepel, a képernyőre kerül kirajzolásra.
* clear\_background() – Minden ciklus elején törli a képernyőt, így előkészíti a következő rajzolási műveletet.
* draw\_rectangle() – Egy téglalap kirajzolását végzi a megadott paraméterekkel (x-koordináta, y-koordináta, szélesség, magasság, szín).
* close\_window() – A fő ciklus befejezése után bezárja az ablakot.

A példából jól látható a Raylib könyvtár egyszerű felépítése és használhatósága. A függvénynevek egyértelműek, a működési elv pedig könnyen átlátható, ami megkönnyíti az alapvető grafikus alkalmazások fejlesztését.

A könyvtár emellett széleskörű bővíthetőséget biztosít: támogatja különböző modellek, képek, textúrák és hangfájlok betöltését is, így a megjeleníthető grafikai elemek mennyisége és összetettsége szinte korlátlan.

### Tapasztalatok

A Python és a Raylib együttes használata kényelmes és egyszerű fejlesztési környezetet biztosít bármilyen grafikai megjelenítést igénylő programhoz.

A könyvtár C és C++ nyelvű verziói nagyobb optimalizálási lehetőséget kínálnak, ezért teljesítménykritikus alkalmazások esetében ezek előnyösebbek lehetnek. A Python előnye ugyanakkor a gyors fejlesztés és az egyszerű szintaxis, amely jelentősen csökkenti a fejlesztési időt.

Az A\* algoritmus megvalósítása a korábbi elméleti ismereteknek köszönhetően hatékonyan valósult meg. A működési elv papíron történő előzetes levezetése segítette az implementáció folyamatát, így a kód megírása már célirányosan történhetett.

A fejlesztés során a legnagyobb nehézséget az állapotok indexelése jelentette. Ennek oka, hogy a kétdimenziós adattárolók elemei [sor index][oszlop index] formában érhetők el, míg a rajzolófüggvények (x koordináta, y koordináta) elrendezést használnak. Ez az eltérés kezdetben zavart okozott az adatok kezelésében, azonban a megfelelő konverziók alkalmazásával a probléma kiküszöbölhetővé vált.

1. A megfelelő rész aláhúzandó. Amennyiben a válasza NEM, akkor a következő bekezdést és táblázatot törölje a dolgozatból. [↑](#footnote-ref-1)