SZÍNKÓD

Olyan fejezet, ami átfogalmazást igényel, egy körvonalat ír le

Szabály: Megfogalmazása után vagy kék flag

vagy flag nélkülivé kell válnia

Nem kiforrott, átfogalmazást igényel(het), felül kell vizsgálni

Szabály: 2 iteráció után kerülhet csak le róla a flag

Plágiumot tartalmaz (Le kell hivatkozni / át kell írni)

Szabály: Hivatkozás, kitörlés vagy teljes átfogalmazásig

NEM kerülhet le róla a flag

Fedlap

Feladat

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat / diplomamunka saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban / diplomamunkámban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 20….................................

...............................................

hallgató aláírása

**KONZULTÁCIÓS NAPLÓ**

Hallgató neve: Neptun kód: Tagozat:

Albert Dávid H1B5EF nappali

Telefon: Levelezési cím (pl: lakcím):

+36309067746 2120 Dunakeszi Toldi utca 47/a

Szakdolgozat / Diplomamunka[[1]](#footnote-2) címe magyarul:

Épületen belüli helymeghatározás

Szakdolgozat / Diplomamunka[[2]](#footnote-3) címe angolul:

Indoor location tracking system

Intézményi konzulens: Külső konzulens:

Prof. Dr. Kozlovszky Miklós

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

|  |  |  |  |
| --- | --- | --- | --- |
| **Alk.** | **Dátum** | **Tartalom** | **Aláírás** |
| **1.** | **2022. 03. 22** | **Technológiai lehetőségek áttekintése,**  **feladatok egyeztetése, ütemterv készítése** |  |
| **2.** | **2022. 04. 05** | **Témán belüli irány meghatározása** |  |
| **3.** | **2022. 04. 20** | **Dokumentum formázásának javítása** |  |
| **4.** | **2022. 05. 02** | **Megvalósítási tervezet, összegzés táblázat**  **javítása és kiegészítése** |  |

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat / Szakdolgozat I. / Szakdolgozat II. / Diplomamunka II. / Diplomamunka III. / Diplomamunka IV. / Projektlabor 2. / Projektlabor 3. / Záródolgozati projekt [[3]](#footnote-4) tantárgy követelményét teljesítette, beszámolóra / védésre [[4]](#footnote-5)bocsátható.

**Javasolt érdemjegy:** ……………………

Intézményi konzulens

Budapest, 2022. 05.13.

Kivonat

Abstract

**Tartalomjegyzék**

[Bevezetés 2](#_Toc147794858)

[1. Elérhető szimulátorok 4](#_Toc147794859)

[1.1. UWSim 4](#_Toc147794860)

[1.1.1. Működése 4](#_Toc147794861)

[1.1.2. Előnyei 5](#_Toc147794862)

[1.1.3. Hátrányai 5](#_Toc147794863)

[1.2. UUVSimulator 6](#_Toc147794864)

[1.3. Aqua Underwater Simulator 6](#_Toc147794865)

[1.4. Stonefish 6](#_Toc147794866)

[1.5. HoloOcean 6](#_Toc147794867)

[2. Szimulátor létrehozására alkalmas szoftverek 7](#_Toc147794868)

[2.1. Unreal Engine 7](#_Toc147794869)

[2.2. Unity 7](#_Toc147794870)

[2.3. Gazebo 7](#_Toc147794871)

# Bevezetés

A tengeralattjárók, az autonóm tengeralattjárók (AUV-k) és más mélytengeri járművek az óceánok, tengerek, folyók és más vízi környezetek titokzatos mélységeiben játszanak fontos szerepet. Ezek a kifinomult gépek és eszközeik nélkülözhetetlen szerepet játszanak a felderítő tevékenységek, tengeri kutatások, környezetvédelem és haditengerészet terén, valamint olyan katasztrófák utáni felderítési és mentési műveletek során, ahol emberi beavatkozás kivitelezhetetlen, vagy igen magas kockázatokkal járna.

Ezen járművek célja olyan feladatok ellátása, amelyre más vízi járművek vagy búvárok beavatkozása nem elégséges vagy lehetséges. Példaként említhetjük a tudományos kutatásokat és környezetvédelmi feladatokat, ahol ezek a gépek a tengerfenék térképezésére, olajszivárgások detektálására és tengeri élőlények megfigyelésére használatosak.

A mélységi vízi járművek izgalmas kihívásokat támasztanak a mérnökök és tudósok előtt, mivel a mélytengerben való működtetés rendkívül komplex és veszélyekkel teli feladat. Ilyen egyedi jellemzők például a mélytengeri nyomásoknak való ellenállás, a korlátozott energiatartalék, a kommunikációs nehézségek, és a pontos navigáció a víz alatti akadályok között. Emiatt ezen járművek tervezése, fejlesztése, tesztelése és üzemeltetése komoly megpróbáltatásokkal jár.

Ezen járművek tervezése és működtetése kritikusan fontos, mivel egy tervezési hiba akár a jármű elvesztéséhez is vezethet, illetve a mélytengeri jármű és a hiba típusától függően emberi életek is veszélybe kerülhetnek. Ezért elengedhetetlen, hogy ezeket a járműveket először szimulációkban és teszteken keresztül vizsgáljuk, mielőtt valós működésükre bíznánk őket. A szimulációk lehetővé teszik számunkra, hogy azonosítsuk és javítsuk a tervezési hibákat, és biztosítsuk a járművek biztonságos és hatékony működését a mélytengeri környezetben.

Fontos megjegyezni, hogy a szimulációk és virtuális környezetek kulcsfontosságú szerepet játszanak az AUV-k és UUV-k fejlesztésében és tesztelésében. A különböző szimulációs megoldások lehetnek segítségünkre a tervezési és tesztelési folyamatok hatékonyabbá tételében, és megkönnyíthetik az AUV-k és UUV-k környezetbarát és hatékony működését. Ebben a kontextusban megvizsgáljuk az elérhető, alkalmas szimulátorokat és mérlegeljük, hogy melyeket érdemes felhasználni, továbbfejleszteni, vagy akár egy teljesen új szimulátort létrehozni a megfelelő tesztelés és fejlesztés érdekében. Az elképzelt szimulátor megalkotása során szükséges figyelembe vennünk a motor vagy szoftverrendszer kiválasztását, amely a legjobban illeszkedik a céljainkhoz és lehetővé teszi a valósághű szimuláció megvalósítását.

Az Óbudai Egyetem BioTech Kutató Központjában zajló tengeralattjáró fejlesztésében veszek részt, elsődlegesen, mint a tengeralattjáróhoz való szimulátor elkészítésében, tesztelésében.

Ebben a bevezetésben áttekintettük a tengeralattjárók, AUV-k és más mélytengeri járművek sokoldalú alkalmazását és az ezekkel járó kihívásokat a vízi környezetek mélységeiben. Emellett hangsúlyoztuk a szimulációk és virtuális környezetek szerepét a járművek tervezési és tesztelési folyamataiban. Meggyőződtünk arról, hogy ezek a technológiák létfontosságúak a mélytengeri járművek fejlesztésében és a tengeri mélységek jobb megértésében és védelmében

Az irodalomkutatást követően a dolgozat bemutatja a szimulátorok kialakítását és működésüket.

# Elérhető szimulátorok

Az első fejezet középpontjában az áll, hogy az UUV-k fejlesztése viszonylag magas költségekkel jár, és a véletlen sérülés vagy elsüllyedés az UUV-k kísérletei vagy tengeri próbái során jelentős veszteségeket okozhatnak. Ebből kiindulva hangsúlyozzuk a hatékony és felhasználóbarát szimulációs rendszerek szükségességét, amelyek segítik az algoritmusok és funkciók ellenőrzését még a tényleges kísérletek vagy tengeri próbák előtt [10,11].

A fejezetben kifejtésre kerül az UUV-k szimulációjának főbb területeit. Emellett bemutatásra kerülnek a már rendelkezésre álló, viszonylag kiforrott szimulációs rendszereket, mint például a UWSim, UUVSimulator, Aqua Underwater Simulator, Stonefish és HoloOcean. Ezekre a szimulátorokra összpontosítva részletes elemzést végzek, kihangsúlyozva előnyeiket és korlátaikat az UUV-k tervezése és tesztelése során.

## UWSim

UWSim, röviden az UnderWater SIMulator, egy olyan szimulációs rendszer, amely a tengeri robotika kutatás és fejlesztés területén használható. Ennek a szoftvernek a segítségével vizualizálható egy víz alatti virtuális scenárió, amelyet standard modellező szoftverek segítségével konfigurálhatunk. Az UWSimbe hozzáadhatunk irányítható víz alatti járműveket, felszíni hajókat, robotikai manipulátorokat és szimulált érzékelőket, amelyeket külsőleg elérhetünk ROS (Robot Operating System) interfészek segítségével. Az UWSim sikeresen alkalmazható víz alatti beavatkozási feladatok logikájának szimulálására és valós beavatkozási feladatok reprodukálására a rögzített naplók alapján.

Az UWSim fejlesztése eredetileg azért indult, hogy egy olyan eszközt nyújtson, amellyel lehetőség van az érzékelési és irányítási algoritmusok tesztelésére és integrálására, mielőtt azokat valós robotokon futtatnánk. Az UWSim fejlesztését az RAUVI és TRIDENT kutatási projektekhez kapcsolódóan indították el.

Az UWSim az alapvető modellező szoftverek segítségével konfigurálható, és tesztelése során Ubuntu Linux rendszereken, 9.10-től 12.04-ig terjedő verziókon használták. Bár nem tesztelték MacOS és Windows rendszereken, az összes használt könyvtár ezekre a platformokra is elérhető, így az UWSim valószínűleg ezeken a rendszereken is működhet.

### Működése

Az UWSim szoftver a vízalatti fizikai szimulációkhoz az Open Dynamics Engine (ODE) nevű fizikai szimulációs motort alkalmazza. Az ODE egy nyílt forráskódú fizikai szimulációs motor, amely C++ nyelven íródott, és rugalmas 3D motorral rendelkezik, amely a jelenet-orientált fejlesztési folyamatokat támogatja. Ennek eredményeként lehetővé teszi a fejlesztők számára, hogy hatékonyan dolgozzanak 3D hardverrel, így könnyebbé és intuitívabbá téve a vízalatti játékok és bemutatók készítését.

Az UWSim jelenetek konfigurálásához XML-formátumú dokumentumokat használ, amelyeket egy DTD dokumentum validál. Az XML fájlok szerkesztésének körülményességét és hatékonyság hiányát elkerülendő, az Xacro makrókat ajánlja a készülékek, objektumok és járművek könyvtárainak előzetes létrehozásához. Ezáltal minimalizálja az XML fájlok időigényes és eredménytelen szerkesztését, miközben elősegíti az elemek újrafelhasználhatóságát és a fejlesztés hatékonyságát.

Az UWSim (UnderWater SIMulator) olyan szoftver, amelyet vízalatti robotika kutatás és fejlesztés során használnak. Az alábbi technológiai megoldásokat alkalmazza:

* Vízalatti jelenetek konfigurálására XML-formátumú dokumentumok segítségével, amelyeket egy DTD dokumentum validál.
* Xacro makrók használatára a hatékonyabb XML-fájlok létrehozásához és könyvtárak kialakításához, minimalizálva az ismétlődő szerkesztést és elősegítve az újrafelhasználhatóságot.
* Óceán jellemzőinek beállítására az "oceanState" blokk segítségével, meghatározva a víz fizikai tulajdonságait a szimuláció során.
* Szimulátor beállítások módosítására a "simParams" blokkban, például shader-ek, felbontás vagy fizikai paraméterek tekintetében.
* Fő kamera paramétereinek beállítására a "camera" blokk segítségével, amely az UWSim főablakában megjelenő jelenetet figyeli és rendeli megjelenítését.
* Vízalatti robotok és szenzorok létrehozására és konfigurálására a "vehicle" címkékkel.
* 3D modellek beillesztésére a jelenetbe és azok interakciójára a "object" blokk használatával.
* ROS interfészek csatolására bizonyos objektumokhoz, robotokhoz vagy szenzorokhoz, egyszerűsítve ezzel a kommunikációt külső vezérlési rendszerekkel.

### Előnyei

* Nyílt forráskódú projekt
* ROS-al összekapcsolható
* Meglévő szimulált járművekkel és környezetekkel rendelkezik

### Hátrányai

* Régi technológiákon alapul
* Környezet telepítési és futtatási nehézségek
* Hiányos, nagyon alapszintű dokumentáció
* Vizuálisan visszamaradott
* Közösségi létszáma alacsony
* Megszűnt a támogatása, fejlesztése (2013)

## UUVSimulator

## Aqua Underwater Simulator

## Stonefish

## HoloOcean

# Szimulátor létrehozására alkalmas szoftverek

Ebben a fejezetben néhány olyan szoftvert tekintünk át, amelyek alkalmasak szimulátorok létrehozására, és amelyek rendelkeznek fizikai motorral, valamint lehetőséget kínálnak az alkalmazások fejlesztésére és testreszabására. A két legkiemelkedőbb platform, amelyeket megvizsgálunk, az Unreal Engine (5) és a Unity. Ezek a szoftverek széles körű eszközöket és funkciókat biztosítanak a szimulátorok létrehozásához és testreszabásához, és lehetőséget kínálnak a valósághű fizika modellezésére.

Emellett tekintetbe vesszük további olyan szimulációs platformokat is, amelyek szintén alkalmasak saját szimulátorok készítésére, például a Gazebo vagy a CARLA. Ezek a szoftverek is lehetőséget biztosítanak az alkalmazások testreszabására és fejlesztésére és specifikus igények kielégítésére is használhatók.

A fejezet során áttekintjük ezeket a szoftvereket, bemutatjuk az előnyeiket és korlátaikat, és segítséget nyújtunk a megfelelő platform kiválasztásában annak érdekében, hogy hatékonyan fejleszthessünk saját szimulátort a céljaink eléréséhez.

## Unreal Engine

## Unity

## Gazebo

1. Megfelelő aláhúzandó! [↑](#footnote-ref-2)
2. Megfelelő aláhúzandó! [↑](#footnote-ref-3)
3. Megfelelő aláhúzandó! [↑](#footnote-ref-4)
4. Megfelelő aláhúzandó! [↑](#footnote-ref-5)