

SVEUČILIŠTE U RIJECI  
**TEHNIČKI FAKULTET**  
Diplomski studij računarstva

Diplomski rad

**Upravljanje robotom i mapiranje okoline  
u Unity 3D**  
**(Robot control and mapping with Unity  
3D)**

Rijeka, rujan 2020.

Aleks Marković  
0069069268

SVEUČILIŠTE U RIJECI  
**TEHNIČKI FAKULTET**  
Diplomski studij računarstva

Diplomski rad

**Upravljanje robotom i mapiranje okoline  
u Unity 3D  
(Robot control and mapping with Unity  
3D)**

Mentor: prof.dr.sc. Kristijan Lenac

Rijeka, rujan 2020.

Aleks Marković  
0069069268

Umjesto ove stranice umetnuti zadatak  
za završni ili diplomski rad

## **Izjava o samostalnoj izradi rada**

Izjavljujem da sam samostalno izradio ovaj rad.

Rijeka, rujan 2020.

-----  
Ime Prezime

# Zahvala

Zahvaljujem xxxxxx na podršci tijekom pisanja ovoga rada i korisnim raspravama i savjetima. Zahvaljujem xxxxx na podršku tijekom studiranja.

# Sadržaj

Popis slika	viii
Popis tablica	ix
<b>1 Uvod</b>	<b>1</b>
1.1 Naslov sekcije . . . . .	1
1.2 Naslov sekcije . . . . .	1
<b>2 Softverski alati</b>	<b>2</b>
2.1 ROS . . . . .	2
2.2 Unity . . . . .	3
2.2.1 Editor . . . . .	4
2.2.2 Skripte . . . . .	6
2.3 ROS# . . . . .	7
<b>Bibliografija</b>	<b>9</b>
<b>Pojmovnik</b>	<b>10</b>
<b>Sažetak</b>	<b>11</b>

## *Sadržaj*

<b>A</b>	<b>Naslov priloga</b>	<b>12</b>
A.1	Naslov sekcije . . . . .	12
A.2	Naslov sekcije . . . . .	12

# Popis slika

2.1	Unity Editor . . . . .	5
-----	------------------------	---



# Popis tablica

2.1	ROS 1 i ROS 2 bitne razlike . . . . .	3
-----	---------------------------------------	---

# Poglavlje 1

## Uvod

Tema ovog diplomskog rada je napraviti funkcionalnu aplikaciju za upravljanje robotom i mapiranje okoline koristeći Unity 3D razvojni program. Zahvaljujući Unity-ju biti će lakše ostvariti cilj da se napravi univerzalni i multiplatformski softver s kojim će se moći upravljati s više vrsta robota.

Kao glavni alat za spajanje i upravljanje na robota koristi se ROS (Robotski Operacijski Sustav) 1. Za omogućavanje komunikacije između ROS-a, tj. robota i Unity aplikacije, koristiti ćemo ROS# knjižnicu. Za svrhu implementacije i testiranja kao testnog robota odabran je popularni Turtlebot 3. Konkretnije koristit ćemo simulirano okruženje (simulaciju) Turtlebot-a i njegovog modela..

### 1.1 Naslov sekcije

### 1.2 Naslov sekcije

# Poglavlje 2

## Softverski alati

Prije samog rješavanja problematike kako napraviti navedenu aplikaciju, potrebno je objasniti što su i kako funkcioniraju korišteni softverski alati. Definirati će se i koji su preduvjeti, tj. knjižnice ili alati koji svaki od njih zahtjeva da se može odraditi funkcija koja im se zada za prethodno navedenu svrhu.

Korištene su najnovije dostupne a stabilne inačice korištenih alata:

1. ROS Noetic Ninjemys - datum izlaska 23. svibnja 2020.
2. Unity 2019. LTS - izašlo polovicom 2020. godine
3. ROS# 1.6 - datum izlaska 20. prosinca 2019.

### 2.1 ROS

Robotski Operacijski Sustav (ROS) je radni okvir (eng. framework) koji se instalira u Linux operacijski sustav i unatoč tome što sadrži riječi operacijski sustav, on to nije. Postoji i eksperimentalna verzija za Windows 10 i OS X, no ovaj će se rad usredotočiti na razvoj na Linux-u. Jedna od najbitnijih karakteristika ROS sustava jest da je omogućena komunikacija i upravljanje hardverom robota preko softverskih alata ROS-a bez da se treba imati posebno znanje o korištenom hardveru.

ROS se ponajviše koristio u znanstvene i obrazovne svrhe, ali se zbog svoje praktičnosti i potencijala ubrzo proširio i u ostale grane robotike. Prije prelaska na

## Poglavlje 2. Softverski alati

ROS, svaki proizvođač robota je je trebao razvijati svoj API (Application Programming Interface) za komunikaciju i upravljanje svojim robotima. Sada roboti diljem svijeta većinom koriste ROS kao svoj primarni sustav za komunikaciju i upravljanje, te je zbog toga vrlo korisno naučiti ROS. Sa istim znanjem i vještinama moguće je razvijati softver koji će poslužiti na različitim robotima, različitim proizvođača, upravo radi ROS unificiranja.

ROS sadrži razne alate i knjižnice, koji su razvijeni i posloženi po određenoj ROS konvenciji. Sve zajedno jako pojednostavljuje razvoj novih robotskih softvera i omogućava kompleksno ponašanje robota. Također ROS sadrži razne upravljačke programe i algoritme, i sve je otvorenog koda - besplatno.[1]

Nedavno je razvijen i novi ROS, ROS 2. Prva službena verzija izdana je krajem 2017. Iako je preporučljivo koristiti više future-proof tehnologije, za ovaj rad odabran je ROS 1 iz razloga što neki od kritičnih alata za uspjeh rada nisu podržani na najnovijim verzijama ROS-a 2. Unatoč tome što je ROS 2 više future-proof i noviji, ROS 1 se i dalje razvija te nova verzija izlazi svake godine. Neke od razlika između ROS-a 1 i 2 navedene su u tablici 2.1.

Tablica 2.1 ROS 1 i ROS 2 bitne razlike

Značajka	ROS 1	ROS 2
Testirane platforme	Ubuntu	Ubuntu, OS X, Windows 10
C++	C++11	C++14, C++17
Python	Python 2	Python 3.5
Roslaunch	XML	Python - kompleksnija logika
Čvor u procesu	1	Više od 1
Ostalo	Velika zajednica s puno stabilnih i odličnih paketa. Puno literature i tutorijala.	Minimalne ovisnosti, bolja prenosivost, pouzdanost, perzistentnost i rad u stvarnom vremenu (real-time).

## 2.2 Unity

Unity je cross-platform (multi-platformsko) razvojno okruženje, primarno za razvijanje računalnih igara. Sveukupni razvoj Unity razvojnog okruženja radi američka korporacija Unity Technologies. Također, sam softver je besplatan za edukacijske i

osobne svrhe te komercijalne svrhe do 100.000 američkih dolara prihoda. Postoji veliki izbor besplatnih i ne besplatnih paketa koji se može koristiti za ubrzati i olakšati određene zadatke, a to se sve nalazi na tzv. "Unity Asset Store". [2]

Unity LTS (Long Term Support) verzija se u tekućoj godini dovrši za prošlu - Unity 2019. LTS je došao sredinom 2020. godine, a nova, 2020. verzija je već dostupna i ona se postepeno nadograđuje. LTS verzija se svakog tjedna ažurira novim zakrpama.

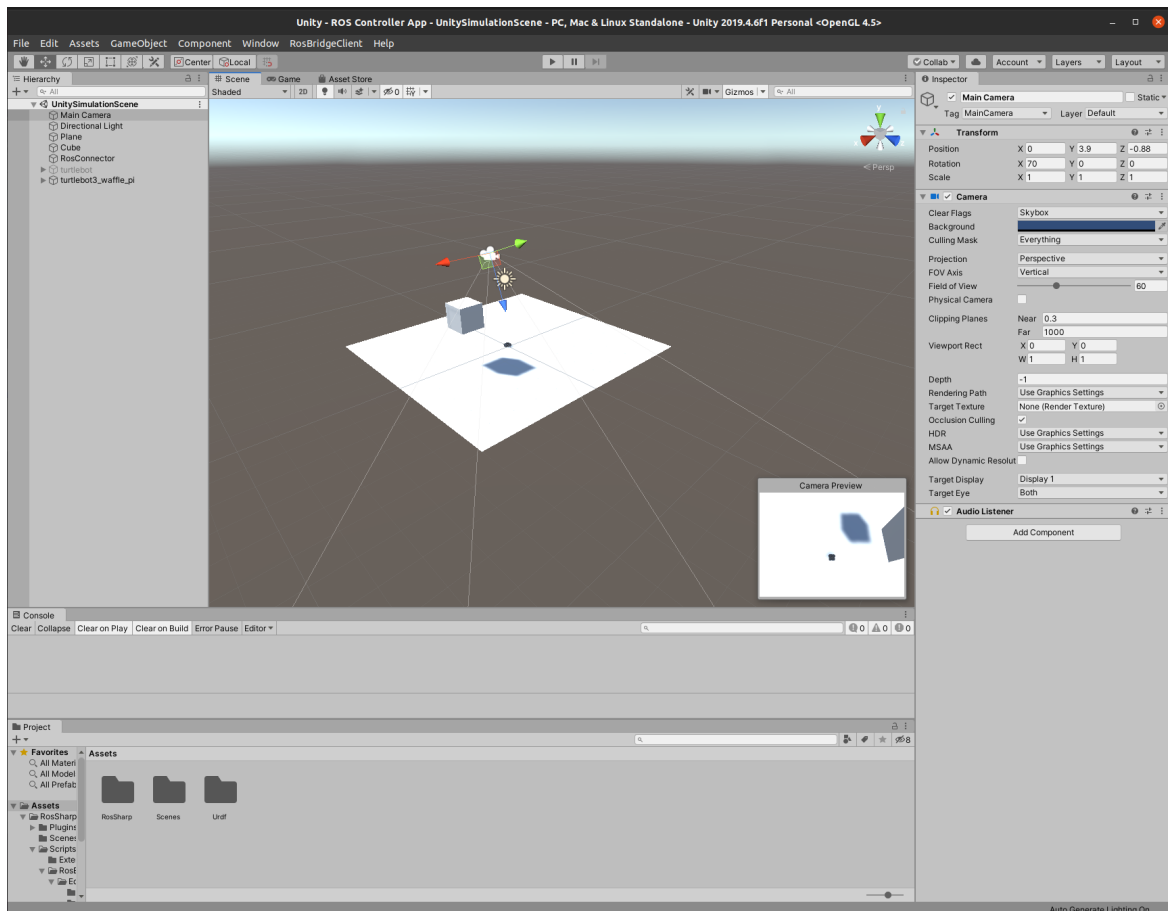
### **2.2.1 Editor**

Unity Editor je glavni alat za razvoj (slika 2.1). U njemu se radi većina razvoja oko vizualnih elemenata i interakcije između njih. Glavne komponente uređivača su sljedeće:

- Hierarchy (hijerarhija) - Hijerarhijski prikaz svih elemenata u trenutnoj sceni. Unity svoje elemente u sceni naziva GameObject (igrači objekt).
- Scene (scena) - Glavni prozor gdje se radi sve u vezi s elementima koji se mogu grafički prikazati.
- Game (igra) - Prozor koji služi kao emulacija igre. Pritiskom na "Play" gumb pokreće se igra i ovaj pokreće se ovaj prozor gdje se može igrati i vidjeti što i kako radi nakon izrade u prozoru scene.
- Inspector (inspektor) - Nakon odabira određenog elementa iz hijerarhije (element se također može odabrati i u sceni), ovdje se prikazuju sve komponente nadodane na taj odabrani element tj. GameObject. Mogu se dodavati razne komponente koje Unity pruža, tipa prikaz slika i teksta, animacije, efekti i još mnogo toga, ali i skripte koje sama osoba koja razvija softver može napisati.
- Console (konzola) - Mjesto gdje se ispisuju sve poruke, upozorenja i greške.
- Project (projekt) - Ovime se može upravljati arhitekturom projekta. Od klasičnog stvaranja mapa i datoteka, imenovanja i ostalog, do dodavanja specifičnih Unity dodataka.
- Ostalo - Meni traka u kojoj se mogu pristupiti svim ostalim postavkama (uključujući

## Poglavlje 2. Softverski alati

i već navedenim stavkama). Dodatni alati se mogu postaviti kao aktivni i vidljivi, npr. pristup Asset Store-u, panel za animacije i animator te ostale stvari koje nisu vezane za zadatak ovog rada.



Slika 2.1 Unity Editor

## 2.2.2 Skripte

Unity također svojim korisnicima omogućava programiranje vlastitih (ili korištenje tuđih) skripta. Programiranje se vrši u C# programskom jeziku, koji se izvršava na *Mono* razvojnom okruženju. Također, moguće je skripte programirati u *JavaScriptu*, ali je podrška lošija.

Cilj *Mono* razvojnog okruženja je da omogući korištenje .NET Framework-a na razne operacijske sustave, jer je inače .NET framework razvijen od strane Microsofta samo za Windows OS. Komponente *Mono* framework-a uključuju:

- C# kompajler - Potpun kompajler, sa svim značajkama od C# 1.0 do 6.0.
- Mono Runtime [3] - Glavne komponente:
  - Just-in-Time (JIT) kompajler - kompilacija koda tokom izvršavanja programa.
  - Ahead-of-Time (AOT) kompajler - kompilacija koda u nativni kod stroja gdje će se izvršavati, prije njegovog izvršavanja.
  - Čitač knjižnica - omogućuje nam učitavanje vanjskih programskih knjižnica.
  - Sakupljač smeća (Garbage collector) - automatsko brisanje objekata u memoriji koji se više ne koriste.
  - Dretveni sustav - omogućava izvršavanje više dretva (threads) istovremeno, pospješuje brzinu i produktivnost određenog programa ako je isprogramirano na pametan način.
  - Interoperabilnost - karakteristika da više programskih sustava mogu bez poteškoća međusobno funkcionirati.
- .NET Framework knjižnica klasa - *Mono* platforma pruža set klasa kao temelj za razvijanje aplikacija. Navedene su klase kompatibilne s Microsoftovim .NET Framework klasama.
- *Mono* pruža i dodatne klase koje nisu uključene u Microsoftovim baznim klasama koje su posebice korisne za razvijanje Linux aplikacija, npr. Gtk+, Zip, OpenGL, Cairo, POSIX i druge.

## Poglavlje 2. Softverski alati

Glavne značajke *Mono* radnog okruženja:

- Multi-platformsko radno okruženje - Linux, macOS, BSD, Microsoft Windows.
- Multi-jezičan - C#, VB 8, Java, Python, Ruby i još mnogi.
- Kompatibilan s binarnim kodom.
- Kompatibilan s Microsoft APIjem - pokreće ASP.NET, ADO.NET, Silverlight i Windows.Forms.
- Otvorenog koda i besplatan - sav *Mono* sadržaj, uključujući i knjižnice, distribuiran je koristeći MIT licencu.
- Opsežna pokrivenost - implementacije mnogih popularnih knjižnica i protokola.

Programiranje prilagođenih skripti omogućava kompleksna ponašanja i radnje u igri/softwareu kojega se razvija. Gotovo sve akcije koje se može napraviti u Editoru, može se i u skripti, te se time može drastično povećati opseg mogućnosti.

---

```
#include <stdio.h>

int main(int argc, char ** argv)
{
    printf("Hello world!\n");
    return 0;
}
```

---

### 2.3 ROS#

ROS# je set programskih knjižnica i alata otvorenog koda u C#-u koja omogućuje komunikaciju između ROS-a i .NET aplikacija - Unity-ja, razvijen od strane Siemens kompanije. ROS# je bio primarno razvijen za Windows OS, ali se uspješno može koristiti i na druge operacijske sustave. Paket se može preuzeti s Unity Asset Store-a ili direktno sa službenog Github repozitorija. [4] Ovo je ujedno i glavni alat koji će se koristiti u ovom radu za stvaranje mosta između ROS-a i Unity-a.

Glavni sadržaj ovog paketa je sljedeći:



## *Poglavlje 2. Softverski alati*

- .NET rješenje za RosBridgeClient (knjižnica koja omogućuje spajanje vanjskih aplikacija na ROS sustav), Urdf (robotski modeli) i MessageGeneration (generiranje poruka).
- ROS paketi korišteni od strane ROS#-a.
- Unity projekt koji sadrži primjer scenu te RosBridgeClient, Urdf i MessageGeneration ekstenzije za Unity.

# Bibliografija

- [1] Ros službene stranice. , s Interneta, [www.ros.org](http://www.ros.org) , 17. rujna 2020.
- [2] U. Technologies. Unity asset store. , s Interneta, <https://assetstore.unity.com> , 22. rujna 2020.
- [3] M. Project. Mono. , s Interneta, <https://www.mono-project.com/docs/about-mono/> , 21. rujna 2020.
- [4] Siemens. Ros#. , s Interneta, <https://github.com/siemens/ros-sharp> , 23. rujna 2020.



# Sažetak

Ovo je tekst u kojem se opiše sažetak vašega rada. Tekst treba imati duh rekapitulacije što je prikazano u radu, nakon čega slijedi 3-5 ključnih riječi (zamijenite dolje postavljene općenite predloške riječi nekim suvislim vlastitim ključnim riječima).

***Ključne riječi*** — ključna riječ 1, ključna riječ 2, ključna riječ 3

## Abstract

This is a text where a brief summary of your work is outlined. The text should have a sense of recap of what was presented in the thesis, followed by 3-5 keywords (replace the general keyword templates below with some meaningful keywords of your own) .

***Keywords*** — keyword 1, keyword 2, keyword 3

# Dodatak A

## Naslov priloga

A.1 Naslov sekcije

A.2 Naslov sekcije