VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

VIRTUÁLNÍ PROSTŘEDÍ PRO SIMULACI V MOBILNÍ ROBOTICE

VIRTUAL ENVIRONMENT FOR SIMULATION IN MOBILE ROBOTICS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Josef Chytrý**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Gábrlík, Ph.D.**

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Josef Chytrý ***ID:*** 230083

Ročník: 3 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Virtuální prostředí pro simulaci v mobilní robotice

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvořit 3D virtuální prostředí pracoviště UAMT pro účely simulací v mobilní robotice. Řešení bude využívat framework ROS2, simulátor Gazebo a operační systém Linux (Ubuntu).

1. Seznamte se s nástroji ROS2 a Gazebo a na systému Ubuntu se je naučte používat.
2. Seznamte se s tvorbou modelů světů v nástroji Gazebo a vyzkoušejte si tvorbu modelu.
3. Vytvořte hrubý model interiéru části pracoviště UAMT dle pokynů vedoucího.
4. Model doplňte o detaily pro dosažení vizuální autentičnosti.
5. Otestujte funkčnost simulace pomocí virtuálního robotu ve spolupráci s vedoucím.

DOPORUČENÁ LITERATURA:

PYO, YoonSeok, HanCheol CHO, RyuWoon JUNG a TaeHoon LIM. ROS Robot Programming. Republic of Korea: ROBOTIS Co., 2017. ISBN 979-11-962307-1-5.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 22.5.2024

Vedoucí práce: Ing. Petr Gábrlík, Ph.D.
Konzultant: Ing. Miloš Cihlář

Ing. Miroslav Jirgl, Ph.D.
předseda rady studijního programu

Abstrakt

Náplní této bakalářské práce je seznámení se s nástroji ROS2 a Gazebo, využívanými na operačním systému Ubuntu. Dále je v rámci této práce podrobně popsán postup vytváření referenčního modelu kostky a modelu prostředí pro simulační platformu Gazebo. Následně je zde rovněž popsán proces vytvoření pracoviště UAMT. Dále jsou v práci popsány možnosti optimalizace simulace. Nakonec je v práci specifikován postup propojení softwaru Gazebo s frameworkem ROS.

Klíčová slova

ROS2, Gazebo, Solidworks, Blender, SDF, STL, DAE, XML, LiDAR, Rviz

Abstract

The purpose of this bachelor's thesis is to get acquainted with the ROS2 and Gazebo tools used on the Ubuntu operating system. Additionally, the thesis provides a detailed description of the process for creating a reference model of a cube and an environment model for the Gazebo simulation platform. It also describes the process of creating the UAMT workplace. Furthermore, the thesis discusses the possibilities for simulation optimization. Finally, it specifies the procedure for integrating the Gazebo software with the ROS framework.

Keywords

ROS2, Gazebo, Solidworks, Blender, SDF, STL, DAE, XML, LiDAR, Rviz

Bibliografická citace

CHYTRÝ, Josef. Virtuální prostředí pro simulaci v mobilní robotice [online]. Brno, 2024 [cit. 2024-05-20]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/160058. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Petr Gábrlík.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Josef Chytrý*
VUT ID studenta:	*230083*
Typ práce:	*Bakalářská práce*
Akademický rok:	*2023/24*
Téma závěrečné práce:	*Virtuální prostředí pro simulaci v mobilní robotice*

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 21. května 2024 ----------

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Petrovi Gábrlíkovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce. Dále děkuji odbornému konzultantovi Ing. Milošovi Cihlářovi za metodickou a odbornou pomoc při zpracování praktické části mé bakalářské práce.

V Brně dne: 21. května 2024 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Moderní vývoj v oblasti mobilní robotiky a automatizovaných systémů vyžaduje neustálý pokrok v technologických dovednostech a schopnostech simulací. Tématem této práce je vytvoření 3D virtuálního prostředí, které bude sloužit k simulacím v mobilní robotice. Zabývá se vývojem a implementací prostředí, ve kterém lze testovat a optimalizovat chování robotů bez nutnosti spouštění fyzických zařízení. Tímto způsobem lze šetřit čas a zároveň umožnit testování různých podmínek a kooperaci robotů.

Cílem této práce je dosáhnout plnohodnotné simulace pracoviště UAMT, umožňující testování a vývoj mobilních robotů v bezpečném a efektivním virtuálním prostředí s využitím moderních technologií, jako jsou ROS2, Gazebo, Solidworks a Blender.

Obsahem práce je seznámení se s nástroji ROS2 a Gazebo, spolu s jejich praktickým použitím v prostředí operačního systému Ubuntu. Dále vytvoření jednoduchého modelu, který je následně implementován do simulačního prostředí Gazebo. Součástí práce je také vytvoření hrubého modelu interiéru pracoviště UAMT, který je doplněn o výbavu místností a chodby. V další fázi jsou vytvořené světy s různými konfiguracemi, které jsou testovány a optimalizovány pro zajištění plynulosti a efektivnosti simulace. V neposlední řadě je vytvořen hrubý model mobilního robota. Robot je testován ve světě a získaná data ze senzoru LiDARu a kamery jsou vizualizována v nástroji Rviz.

1. SOFTWAROVÉ NÁSTROJE

1.1 ROS2

Robot Operating System (ROS2) představuje kolekci volně přístupných softwarových knihoven a nástrojů určených pro vývoj robotických aplikací. Poskytuje komplexní soubor open source prostředků, které zahrnují ovladače, moderní algoritmy a výkonné nástroje pro vývoj.

ROS2 zpracovává informace v takzvaných nodech (uzlech), které mohou odesílat a přijímat data z jiných uzlů prostřednictvím topics (témat) services (servisů), actions (akcí) nebo parametrs (parametrů). Nody jsou rozděleny na Publisher a Subscriber. Slouží pro sdílení informací mezi jednotlivými nody. Publisher data odesílá a Subscriber je přijímá. Tím se zajistí efektivní a asynchronní komunikace mezi různými částmi systému. [1]

Verze ROS je rozdělená do takzvaných distribucí. Tato práce využívá aktuální distribuci Humble Hawksbill.

1.2 Gazebo

Gazebo je kolekce volně přístupných softwarových knihoven navržená pro zjednodušení vývoje vysoce výkonných aplikací. V této práci je použita knihovna Simulation, která dokáže simulovat prostředí a robotické systémy pro výzkum, vývoj a návrh. [2] Tato práce využívá verzi Gazebo Garden.

1.3 Solidworks

Solidworks představuje 3D CAD software, který je uplatňován převážně ve strojírenském průmyslu. V této práci je využíván pro objemové modelování, například obvodových zdí, nábytku a podlahy. [3]

Důvodem výběru právě tohoto softwarového nástroje bylo snadné a vizuálně přehledné ovládání a možnost importování modelu ve formátu, který software Blender podporuje.

1.4 Blender

Blender je bezplatná open source (volně přístupná) sada nástrojů pro tvorbu a práci s 3D objekty. Podporuje celou řadu nástrojů od modelování přes simulaci až po tvorbu her. [4]

Blender byl využit k přidání textur na konkrétně vytvořený model v Solidworks pro zajištění realističnosti simulovaného prostředí.

2. POSTUP TVORBY MODELU PRO GAZEBO

2.1 Vytvoření Gazebo světa pro simulaci

Pro zobrazení světa v Gazebu je zapotřebí soubor formátu SDF.

2.1.1 SDF soubor

Simulation Description Format, zkráceně SDF, je formát XML jazyka, který slouží k popisu objektů a prostředí pro simulátory robotů, vizualizaci a řízení. Původně byl součástí simulování robotů v Gazebo. Vývojem a stálým zdokonalováním je schopný podrobně popsat všechny aspekty robotů, statické i dynamické objekty, osvětlení, terén, a dokonce fyzikální vlastnosti. [5]

2.1.2 Základní definování světa SDF

Každý SDF svět musí začínat s těmito elementy:

```
<?xml version="1.0" ?>
<sdf version="1.8">
    <world name="demo_cube">
    ...
    </world>
</sdf>
```

První dva řádky jsou definice verze XML jazyka a SDF. Svět je potřeba pojmenovat. Mezi elementy <world> se následně zahrnuje veškerý kód. [6] Název souboru je například *cube-world.sdf*.

2.1.3 Nastavení fyzikálních vlastností

Příklad XML kódu:

```
<gravity>0 0 -9.8</gravity>
<physics default="0" name="default_physics" type="ode">
      <max_step_size>0.001</max_step_size>
      <real_time_factor>1</real_time_factor>
      <real_time_update_rate>1000</real_time_update_rate>
</physics>
```

Pro správné fungování simulace je nezbytné definovat klíčové fyzikální vlastnosti, jako jsou gravitační síla a směr jejího působení. Toto nastavení je implementováno pomocí elementu <gravity>, kde jsou velikosti a směr působení gravitace numericky definovány v kartézském souřadnicovém systému (x, y, z). Dále je v konfiguraci fyzikálního enginu specifikován typ systému, konkrétně ode (open dynamics engine), což ovlivňuje způsob výpočtu fyzikálních interakcí v simulaci. Element <max_step_size> specifikuje maximální časový krok simulace, přičemž hodnota 0.001

značí aktualizaci jednou za milisekundu. Element <real_time_factor> určuje, jak rychle simulace probíhá ve srovnání s reálným časem; hodnota 1 znamená, že simulace běží v reálném čase. Element <real_time_update_rate> udává frekvenci aktualizací simulace, a hodnota 1000 znamená, že aktualizace probíhá každou milisekundu. [6]

2.1.4 Osvětlení světa

Pro dosažení co nejvěrohodnějších detailů textur, je klíčové adekvátně nastavit parametry osvětlení. Níže je uveden příklad konfigurace osvětlení v XML formátu:

```
<light type="directional" name="sun">
    <cast_shadows>true</cast_shadows>
    <pose>0 0 9 0 0 0</pose>
    <diffuse>0.5 0.5 0.5 1</diffuse>
    <specular>0.2 0.2 0.2 1</specular>
    <attenuation>
        <range>80</range>
        <constant>0.3</constant>
        <linear>0.01</linear>
        <quadratic>0.001</quadratic>
    </attenuation>
    <direction>0.1 0.1 -1</direction>
</light>
```

Element <pose> určuje pozici (x, y, z) a orientaci (roll, pitch, yaw) světelného zdroje. Vlastnosti <diffuse> a <specular> mají zásadní vliv na celkovou kvalitu osvětlení. Difuzní komponent <diffuse> udává míru, jakým je světlo rozptýleno na povrchu materiálu do všech směrů, zatímco spekulární komponent <specular> ovlivňuje odraz světla z lesklých povrchů pod specifickým úhlem. [6] Vyšší hodnoty spekulárního odrazu způsobí, že objekty s lesklými povrchy, jako jsou kovy, se budou více lesknout.

Element <attenuation> popisuje způsoby útlumu světla. Subelement <range> definuje dosah světla, <constant> ovlivňuje celkovou intenzitu světla nezávisle na vzdálenost, <linear> lineární útlum světla, který se zvyšuje se vzdáleností a <quadratic> kvadratický útlum dále zvyšuje útlum světla exponenciálně s rostoucí vzdáleností neboli přidává k útlumu zakřivení. Tento útlum je nejsilnější ve větších vzdálenostech. [6]

Směr světla je specifikován pomocí elementu <direction>, což umožňuje přesnou orientaci světelného toku v simulovaném prostředí. [6]

2.2 Vytvoření modelu kostky v Solidworks

Kostka byla zvolena jako referenční model s cílem demonstrovat schopnosti orientace v rámci různých frameworků. Tento model byl preferován pro svoji jednoduchost vytváření Ušetřený čas byl následně využit k průzkumu dalších klíčových aspektů spojených s modelováním a texturováním. Tato rozhodnutí byla podpořena skutečností, že práce v prostředí Blender vyžadovala větší časovou investici. Všechny

vytvořené modely, materiály a kódy jsou systematicky ukládány do repozitáře na platformě GitHub, což umožňuje snadnou správu, další úpravy a sdílení.

2.2.1 Možnosti modelování

Software Solidworks nabízí uživatelům možnost vybrat si před zahájením modelovacího procesu mezi vytvářením dílu, sestavy, nebo výkresu. Tato volba umožňuje sestavit komplexní modely, které jsou integrovány do jednoho finálního virtuálního prostoru uloženého v jediném souboru. Avšak tento přístup může být neefektivní, zejména pokud je později vyžadována úprava rozmístění jednotlivých komponentů sestavy. Z tohoto důvodu je často preferováno modelování jednotlivých dílů zvlášť. Tento postup umožňuje jednotlivým prvkům zaujímat specifické pozice ve světě Gazebo, což usnadňuje budoucí úpravy a integraci do různých scénářů.

2.2.2 Nejčastěji používané nástroje

V procesu modelování se začíná vytvářením 2D skic, které jsou základem pro generování 3D tvarů. Kreslení skic zahrnuje použití nástrojů jako jsou čáry, oblouky a kružnice. Tyto prvky jsou esenciální pro definování základních geometrických tvarů, které jsou následně transformovány do trojrozměrného prostoru. Pro přeměnu 2D skic na 3D objekty je primárně využíván nástroj *Extrude*. Tento nástroj umožňuje vysunutí skicy do prostoru, čímž vzniká plně trojrozměrný model.

2.2.3 Náčrt a vysunutí kostky

Vzhledem k jednoduchosti modelu kostky bylo vytvoření skici o rozměru 1x1 m nenáročným úkolem. Po dokončení této skici byla použita technika extruze, která transformovala dvourozměrný náčrt na plně trojrozměrný objekt. Výsledkem je model kostky s rozměrem jednoho metru na každé straně.

2.2.4 Export modelu kostky

Software Solidworks nabízí robustní možnosti pro export vytvořených modelů do široké škály formátů. Pro 3D objekty, které budou využity v jiných 3D aplikacích je vhodný STL formát. STL neboli Standard Triangle Language popisuje geometrie povrchu objektu, kde je každý segment reprezentován trojúhelníkovými plochami. Díky své struktuře a efektivitě je STL široce používán pro sdílení a tisk 3D modelů, poskytující základ pro další manipulaci a zpracování v různých 3D tiskových a vizualizačních nástrojích. Jeho nevýhodou je, že neobsahuje barvy a textury. [7]

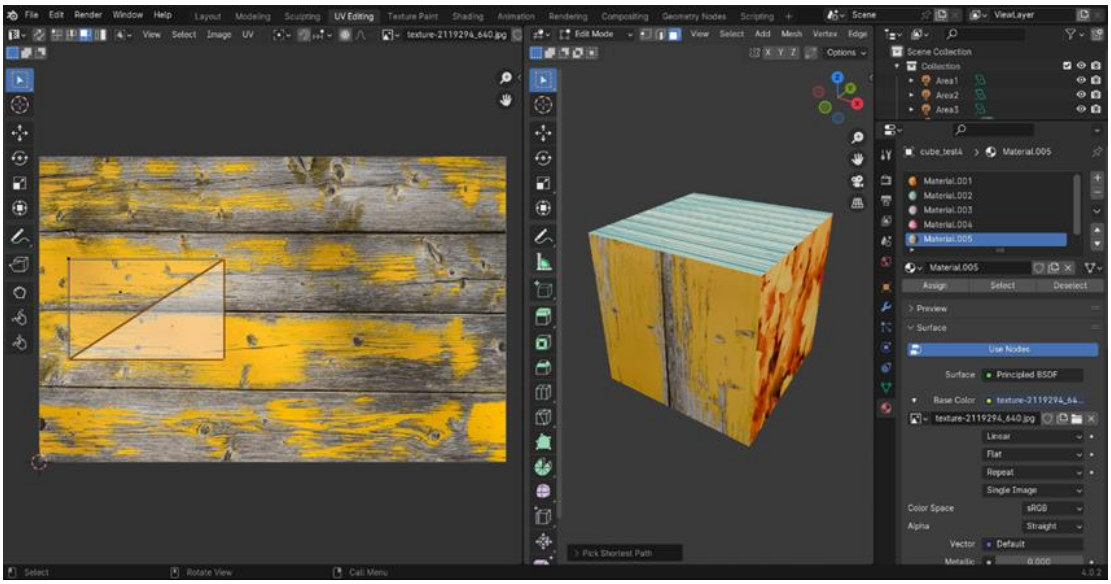
2.3 Texturování modelu kostky v Blenderu

Blender je komplexním softwarovým nástrojem, který má mnoho oficiálních video tutoriálů na platformě Youtube. Tyto tutoriály byly využity k prvním krokům pro práci se softwarem.

2.3.1 Přidání textur modelu

Pro přidání textur k modelu je nejprve nutné model označit. Po kliknutí na ikonu „*světa*“ se otevře záložka s parametry pro nastavení materiálů. Pro přidání textury jakožto obrázku uloženého v počítači, je potřeba kliknout na tlačítko „*Base Color*“ a vybrat možnost „*Image texture*“. Z počítače je následně vybrán uložený obrázek. Poté přes *UV Editor* je možné přiřadit části textury na jednotlivé označené elementy modelu.

Pro přidání více textur na kostku je nutné v záložce s parametry nastavení materiálu kliknout na tlačítko „+“, čímž se vytvoří nové složky s materiály. Do každé složky s materiály je nahrána jedna textura. Je tedy vytvořeno šest různých textur, které je možno přiřadit na konkrétní plochu kostky.



Obrázek 2.1 Ukázka práce v Blenderu. Vlevo je UV Editor, uprostřed se nachází výsledný pohled modelu kostky a vpravo je záložka s přidanými materiály

2.3.2 Export modelu

Podobně jako Solidworks, i Blender nabízí rozsáhlé možnosti pro export a import modelů v různých formátech. Pro účely simulace v Gazebu je model kostky exportován ve formátu DAE (Digital Asset Exchange), který vychází z XML formátu COLLADA (Collaborative Design Activity). Tento formát je speciálně navržen pro podporu výměny dat mezi různými 3D aplikacemi, což usnadňuje interoperabilitu. Model byl pojmenován *demo_cube.dae* a umístěn do složky *meshes*.

2.3.3 Změna adresářové struktury textur v exportovaném modelu

Při exportu modelu dojde k zahrnutí všech použitých textur. Aby byly textury po spuštění modelu v libovolném 3D programu správně zobrazeny, je nutné umístit je do

stejného adresáře jako model. Mé zadání spočívalo v přesunutí těchto textur do nově vytvořené složky nazvané *textures*, čímž se oddělily od hlavního modelu a zvýšila se přehlednost adresáře. V důsledku tohoto přesunu bylo nezbytné upravit cesty k texturám v kódu modelu. Na příslušných řádcích byly cesty k texturám změněny tak, že nyní odkazují o úroveň zpět do složky *textures*. Relativní cesta zpět je označena dvěma tečkami a následuje jméno cílové složky, v tomto případě *textures*. Příklad upraveného XML kódu:

```
<library_images>
    <image id="autumn-83761_640_jpg" name="autumn-83761_640_jpg">
      <init_from>../textures/autumn-83761_640.jpg</init_from>
    </image>
</library_images>
```

2.4 Propojení světa a modelu

Model kostky je uložen ve formátu DAE (COLLADA), který primárně definuje vizuální aspekty modelu, avšak nezahrnuje informace o fyzikálních vlastnostech kostky. Aby bylo možné model integrovat do simulačního prostředí s přesně definovaným chováním, je nezbytné vytvořit soubor *model.sdf*. Tento soubor bude obsahovat veškeré potřebné informace o fyzikálních vlastnostech.

2.4.1 Vytvoření model.sdf

Pro specifikaci fyzikálních vlastností a pozice modelu je vyžadován nový soubor ve formátu SDF. Níže uvedený kód definuje počáteční pozici a orientaci modelu:

```
<pose>0 0 0 0 0 0</pose>
```

Element <pose> určuje, že pro tento příklad je model umístěn ve středu prostoru bez jakékoliv rotace. Vizuální a kolizní reprezentace modelu jsou definovány následujícím způsobem:

```
<visual name="visual">
          <geometry>
            <mesh><uri>model //Cube/meshes/demo_cube.dae </uri></mesh>
          </geometry>
</visual>

<collision name="collision">
          <geometry>
            <mesh><uri>model://Cube/meshes/demo_cube.dae </uri></mesh>
          </geometry>
</collision>
```

Tento segment kódu popisuje vizuální reprezentaci a kolizní geometrii modelu. Je běžné, že vizuální a kolizní modely jsou různé, což umožňuje optimalizaci procesů vizualizace a simulace kolizí. Cesta k modelu začíná od kořenového adresáře //Cube a končí u samotného modelu *demo_cube.dae*, který byl exportován z Blenderu.

Pro definici hmotnostních a inerciálních charakteristik modelu slouží element <inertial>:

```
<inertial>
  <mass>1.0</mass>
          <inertia>
            <ixx>0.003</ixx>
            <iyy>0.0008</iyy>
            <izz>0.004</izz>
          </inertia>
</inertial>
```

Zde <mass> specifikuje hmotnost a <inertia> obsahuje momenty setrvačnosti v osách x, y a z, což jsou klíčové parametry pro realistické chování modelu v dynamickém prostředí. [8]

2.4.2 Vytvoření model.config

Pro každý model vytvořený pro simulace je důležité mít v kořenovém adresáři uložený konfigurační soubor, který obsahuje meta informace. Tyto informace jsou zásadní pro správnou identifikaci a správu modelu a zahrnují jméno modelu, jeho verzi, identifikační údaje autora a krátký popis modelu. Struktura konfiguračního souboru v XML formátu vypadá následovně:

```
<?xml version="1.0"?>
<model>
  <name>Cube</name>
  <version>1.0</version>
  <sdf version="1.5">model.sdf</sdf>

  <author>
    <name>Josef Chytry</name>
    <email>xchytr01@vutbr.cz</email>
  </author>

  <description>
      Cube demo
  </description>
</model>
```

Tento XML dokument začíná deklarací verze <?xml version="1.0"?>, následuje hlavní element <model>, který obsahuje subelementy určující název modelu (<name>), jeho verzi (<version>), cestu k souboru SDF (<sdf>), informace o autorovi (<author>), a krátký popis modelu (<description>). Tento soubor je nezbytný pro integraci modelu do simulace a umožňuje jeho snadnou identifikaci a správu.[9]

2.4.3 Vkládání modelu do světa SDF

Pro integraci vytvořeného modelu do simulovaného světa je nezbytné, aby byl model správně uložen ve složce s modely. Následující XML kód v souboru světa definuje, jak má být model vložen:

```
<model name="Cube_01">
<include>
    <uri>
    model://models/Cube
    </uri>
</include>
<pose>1 -1 0 0 0 0</pose>
</model>
```

Element <include> specifikuje URI adresu modelu, která umožňuje jeho načtení. Atribut *name* v elementu <model> slouží k jednoznačné identifikaci modelu v rámci světa, což je zvláště důležité pro situace, kdy je potřeba do světa vložit více stejných modelů. Každý model musí být jednoznačně pojmenován, aby bylo možné s ním nezávisle manipulovat a přesouvat ho na různé pozice ve světě. Element <pose> určuje počáteční pozici a orientaci modelu ve světě.

2.4.4 Struktura databáze pro funkční simulaci v Gazebo

Pro správnou funkci simulace je klíčové dodržet následující hierarchii složek a souborů:

```
├── models
│   ├── Cube
│   │   ├── meshes
│   │   │   └── demo_cube.dae
│   │   ├── textures
│   │   │   ├── texture1
│   │   │   ├── texture2
│   │   │   ...
│   │   │
│   │   ├── model.config
│   │   └── model.sdf
│   ├── model_2
│   └── ...
└── world
    ├── cube-world.sdf
    ├── world_2.sdf
    ...
```

Ve složce *models* jsou uloženy všechny modely. Každý model, například Cube, má svoji podsložku *meshes*, která obsahuje model ve formátu DAE. K tomuto modelu jsou přiřazeny textury, které jsou uloženy v podsložce *textures*. Složka *world* pak obsahuje

soubory definující svět simulace, včetně *cube-world.sdf*, který integruje modely do simulovaného prostředí. [9]

2.4.5 Simulace v Gazebu

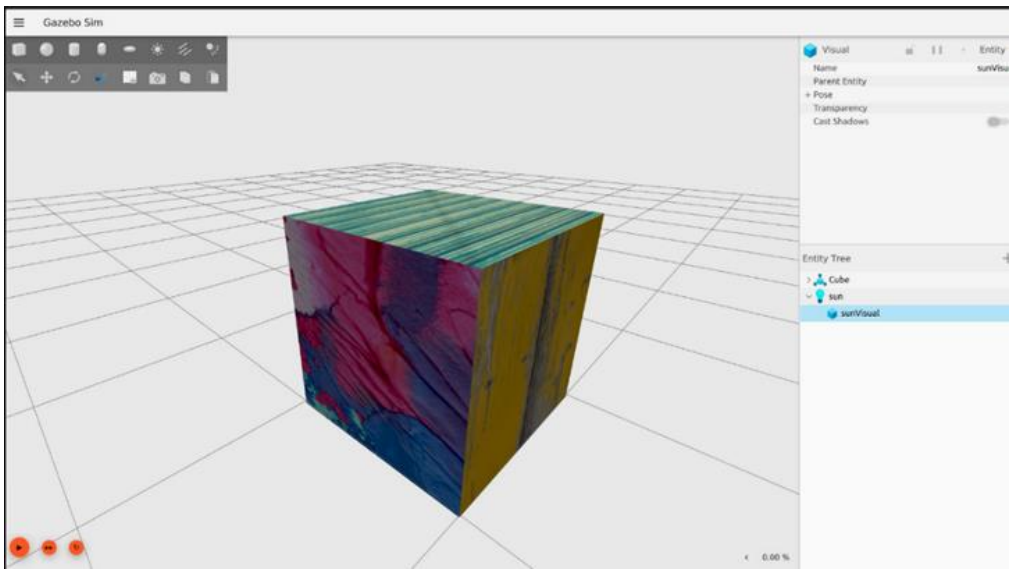
Pro úspěšné spuštění světa v simulačním prostředí Gazebo je nezbytné nejprve nastavit pracovní adresář, kde se nachází soubor *cube-world.sdf*. Tento krok provádíme změnou cílového adresáře pomocí příkazu *cd* v konzoli. Přesun do správného adresáře provedeme následovně:

```
cd ~/world
```

Po změně adresáře spustíme simulaci světa v Gazebu pomocí příkazu:

```
gz sim cube-world.sdf
```

Tento příkaz iniciuje simulaci, v níž je načten definovaný svět ze souboru *cube-world.sdf*, umožňující interakci s modely v simulačním prostředí Gazebo.



Obrázek 2.2 Výsledek simulace modelu kostky v Gazebu

3. MODEL CHODBY A MÍSTNOSTÍ

Hlavním cílem této bakalářské práce je vytvoření 3D virtuálního prostředí pracoviště Ústavu automatizace a měřící techniky (UAMT). V souladu s pokyny vedoucího práce bylo stanoveno, že tři místnosti budou plně vybaveny: laboratoř robotiky – polygon, laboratoř teleprezence a laboratoř mobilních robotů. Zbylé místnosti zůstanou prázdné pro případné budoucí doplnění.

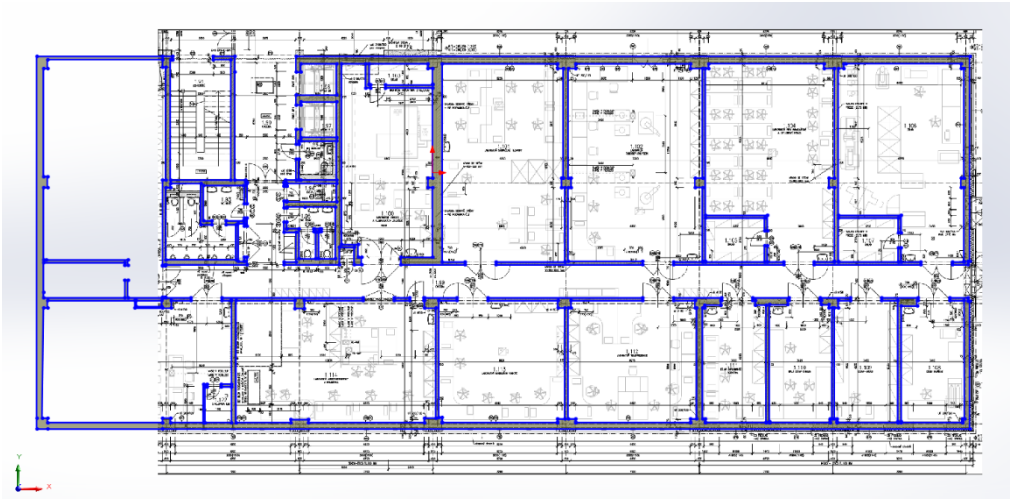
3.1 Tvorba zdí, podlahy a stropu v Solidworks

Vedoucí práce poskytnul výkres půdorysu budovy prvního patra části E ústavu automatizace a měřící techniky. Tento výkres byl oříznut a převeden do formátu *png* pro snadnější práci, a to především kvůli velikosti tohoto souboru.

3.1.1 Postup modelování zdí

Prvním krokem bylo vytvoření skici, přičemž Solidworks nabízí možnost vložení referenčního obrázku pro následné načrtnutí skici. Po vložení oříznutého půdorysu bylo nezbytné definovat měřítko celého modelu. Tento krok byl realizován vybráním libovolné části zdi, na které byla vytvořena čára od začátku do konce profilu zdi. Tato čára byla následně přesně dimenzována na základě kóty vybrané zdi z půdorysu, což zajistilo přesné měřítko modelu v souladu s reálnou velikostí.

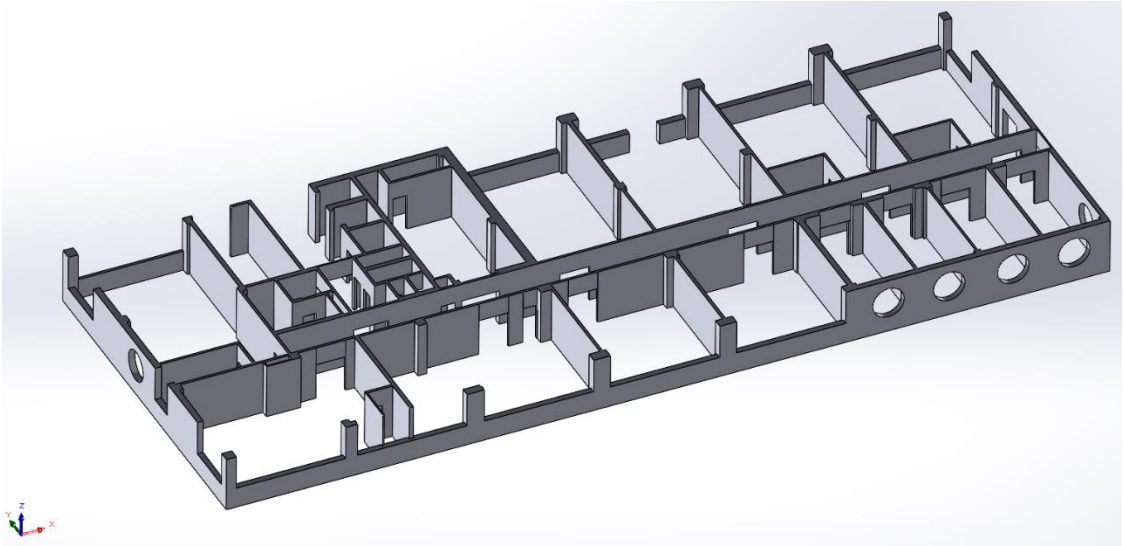
Následujícím krokem bylo vykreslení obvodu vybraných zdí, zahrnující jak venkovní, tak vnitřní hrany. Vytvořená skica byla poté jedním stiskem tlačítka převedena do požadované velikosti výšky zdi.



Obrázek 3.1 Ukázka narýsované skici přes referenční obrázek

3.1.2 Tvorba otvorů do modelu zdí

Prvním krokem bylo identifikování a definování všech typů oken určených pro použití v místnostech. Bylo vybráno pět různých druhů oken, přičemž každý typ měl specifické rozměry. Tyto rozměry byly využity při tvorbě přesných skic na modelu zdí. Pro realizaci otvorů ve zdích byla využita funkce „O*debrání vysunutí*“, která umožňuje odstranit objem modelu vysunutím profilu v určeném směru. Po dokončení otvorů pro okna byla zahájena tvorba otvorů pro dveře. Pro dveře byly zvoleny tři různé velikosti zárubní. Podobně jako u oken, byly i otvory pro dveřní zárubně vytvořeny pomocí skic a funkce „*odebrání vysunutí*“.



Obrázek 3.2 Výsledný model obvodových zdí v Solidworks

3.1.3 Postup modelování podlahy a stropu

Podlaha byla rozdělena na dvě hlavní sekce: jedna sekce představovala podlahu chodby a druhá podlahu místností. Tento rozdělený přístup byl zvolen z důvodu efektivnějšího rozdělení texturovaných ploch, což umožňuje snadnější změny textur v budoucnu, aniž by bylo nutné předělávat textury celé sekce, neboť každá sekce má odlišnou texturu.

V softwaru Solidworks byl načten existující model zdí. S využitím nástroje pro měření byl změřen obvod každé z obou definovaných sekcí. Tyto zjištěné hodnoty byly použity pro návrh skic a následné vysunutí ploch obou sekcí. Tento proces vedl k vytvoření dvou odlišných modelů podlahy: model podlahy chodby a model podlahy místností.

Model stropu byl konceptuálně jednoduchý. Byly naměřeny rozměry délky zdí, a tyto hodnoty byly použity pro načrtnutí skici jednoduchého obdélníku, který byl poté extrudován do prostoru. Tento model stropu je využit v různých simulačních scénářích.

3.2 Vybavení pracoviště UAMT

Modely nábytku a příslušenstvích byli získány následovně. Hlavní části, jako jsou okna, dveře a stoly, byly vytvořeny přímo v softwaru Solidworks. Další modely byly získány z oficiálních stránek Gazebo [10] a také byly staženy volně dostupné modely z internetového zdroje [11].

Pro každý získaný nebo vytvořený model byly připraveny soubory *model.sdf* a *model.config*. V souboru *model.config* byly specifikovány copyright informace pro každý model. Všechny modely spolu s jejich soubory byly systematicky ukládány do složky *models*.

3.2.1 Vkládání modelů na pozici v Gazebo světě

Každý model umístěný do světa Gazebo musel být přesně pozicován vůči středu simulovaného světa. Pro určení přesné lokace každého modelu byl využit „Entity tree“, neboli strom entit modelů, kde měl každý model přiřazenou svoji počáteční pozici. Gazebo umožňuje dynamicky měnit pozici modelu změnou hodnot x, y, z, roll, pitch a yaw, což umožňuje přesné umístění modelů v prostoru. Tyto hodnoty byly použity k upravení počáteční pozice modelů při spuštění simulace, takže se modely zobrazovaly přesně na určených místech.

U každého modelu byl postup shodný. V kódu byly reference na modely organizovány podle místností nebo laboratoří, což vedlo k vytvoření přehledného kódu s modely přesně umístěnými podle svého určení ve světě. Příklad kódu vložených modelů dveří do specifických laboratoří:

```
<!-- Laboratoř vibrací a klimatických zkoušek 1.100 -->
<model name="door_70_close_04">
<include>
    <uri>
    model://models/door_70_close
    </uri>
</include>
      <pose>-1.51 8.538 0 0 0 3.14159</pose>
</model>
```

3.2.2 Osvětlení světa

Pro každou místnost i chodbu byl zvolen zdroj světla. V rozsáhlejších laboratořích bylo přidáno více zdrojů světla, aby bylo dosaženo správného zachycení detailů textur. Klíčovým parametrem, který byl upravován, byl rozsah osvětlení (range). Byl použit bodový typ světla, jehož správné nastavení rozsahu zajišťovalo efektivní osvětlení

místnosti. Příliš velká hodnota rozsahu ve vztahu k velikosti místnosti by vedla k jejímu přesvícení, což by mohlo negativně ovlivnit vizuální autentičnost prostoru. Příklad nastavení světla pro laboratoř teleprezence:

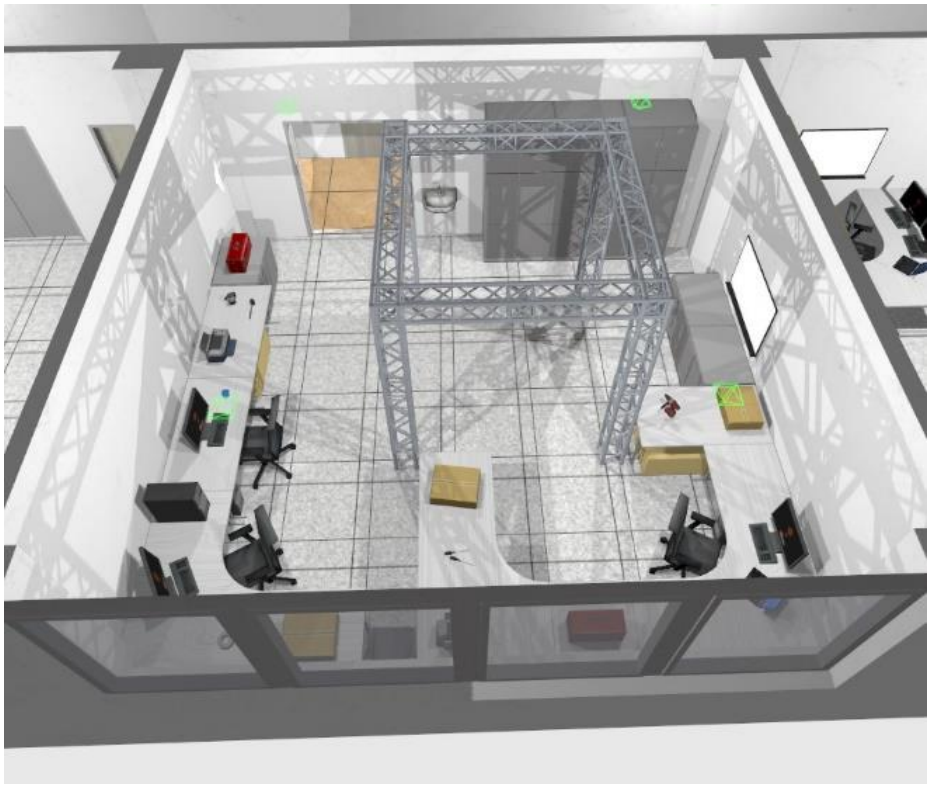
```
<!-- Laboratoř teleprezence 1.112 -->
<light name="ceiling_light_05" type="point">
      <pose frame="">12.5 -6.2 3.21824 0 0 0</pose>
      <diffuse>1 1 1 1</diffuse>
      <specular>1 1 1 1</specular>
      <intensity>0.6</intensity>
      <attenuation>
        <range>6</range>
        <constant>0.3</constant>
        <linear>0.01</linear>
        <quadratic>0.001</quadratic>
      </attenuation>
      <cast_shadows>1</cast_shadows>
      <direction>0.1 0.1 -1</direction>
</light>
```

Teleprezenční laboratoř, která je menší místností bez výrazných překážek ve středu, vyžadovala pouze jeden světelný zdroj pro adekvátní osvětlení. Nastavení rozsahu světla na šest metrů efektivně pokrylo celou místnost s dostatečnou intenzitu.

Obrázek 3.3 Výsledek simulace modelu pracoviště UAMT v Gazebu

Obrázek 3.4 Detailní pohled na laboratoř robotiky-polygon

Obrázek 3.5 Detailní pohled na laboratoř teleprezence



Obrázek 3.6 Detailní pohled na laboratoř mobilních robotů

4. TESTOVÁNÍ A OPTIMALIZACE SIMULACE

Pro praktické využití je zásadní průběh simulace. U některých modelů byla fyzika nastavena na dynamickou, což vedlo k vykonávání nadměrného množství zbytečných výpočtů, způsobujících nefunkčnost celé simulace. Modely s nesprávnými nastaveními byly identifikovány tím, že se v SDF souboru světa nastavila záporná hodnota gravitace. Po spuštění simulace tyto modely začaly stoupat vzhůru.

Tyto modely byly následně upraveny ve svých *model.sdf* souborech. Element <static> byl nastaven na *true*. To znamená, že pokud dojde k nárazu robota do objektu, objekt zůstane na svém místě. Naopak, nastavení tohoto elementu na *false* vyžaduje, aby počítač prováděl složitější výpočty pro dynamiku pohybu modelu v případě kolize, což může v důsledku snižovat výkonnost simulace.

Běh simulace je možné sledovat díky několika parametrům zobrazeným na pravé straně spodní lišty Gazebo. Jedním z hlavních parametrů je RTF (Real Time Factor), který ukazuje poměr mezi dobou trvání simulace a reálným časem a indikuje, jak rychle nebo pomalu simulace probíhá ve srovnání se skutečným světem. [12]

V simulátoru Gazebo se stav prostředí aktualizuje při každé iteraci. Každý krok posune simulaci o pevně stanovený časový interval, ve výchozím nastavení je velikost kroku 1 ms. [12]

4.1 Přidání dynamiky do světa

Vzhledem k tomu, že základní simulace obsahuje pouze statické objekty, byl pro ověření její funkčnosti a plynulosti vytvořen dynamický systém. Tento systém může zahrnovat různé prvky, jako je pohyb člověka, otevírání dveří nebo pohyb robotů. Pro demonstraci dynamiky v simulaci je vytvořen příklad pohybu člověka.

4.1.1 Model člověka

V simulaci Gazeba se animovaný model nazývá „actor“. Jsou dva typy animací, které lze využít odlišně nebo kombinovaně. Animace kostry, u které je relativní pohyb mezi vazbami v jednom modelu a animace trajektorie, při které se všechny klouby modelu pohybují jako jednotný celek po definované trajektorii. Kombinace obou typů animací zajišťuje přirozený pohyb modelu. [13]

Model člověka je dostupný na stránkách Gazeba [10]. Jeho vložení do světa je ukázáno v následujícím kódu:

```
<actor name="actor_01">
  <skin>
  <filename>
      model://models/actor/meshes/walk.dae
  </filename>
  <scale>1.0</scale>
```

```
  </skin>
    <animation name="walk">
    <filename>
         model://models/actor/meshes/walk.dae
    </filename>
    </animation>
</actor>
```

Na modely se nevztahuje gravitace ani kolize modelu. Element <skin> načítá DAE soubor, který specifikuje vzhled modelu. Element <scale> určuje měřítko modelu, v tomto případě 1:1. Pod elementem <animation> se specifikuje, jak se model bude pohybovat. [13] U staženého modelu je k dispozici více možností animací chování, jako například běh (run.dae), sednutí (sit_down.dae) nebo mluvení (talk_a.dae). Pro aplikaci dané animace stačí adresovat příslušný soubor DAE.

4.1.2 Vytvoření skriptované trajektorie

Pro dynamické modely v Gazebo lze definovat specifické trajektorie, přičemž pozice, které model dosáhne v určeném čase, jsou přesně zadány. Gazebo automaticky zajišťuje interpolaci pohybu mezi těmito pozicemi pro plynulost animace. Trajektorii je možné nastavit tak, aby se opakovala, jak ukazuje následující kód: [13]

```
<script>
    <loop>true</loop>
    <delay_start>0.000000</delay_start>
    <auto_start>true</auto_start>
</script>
```

Když je element <loop> nastaven na *true*, trajektorie se bude neustále opakovat. Aby bylo možné trajektorii cyklicky opakovat, je důležité, aby se počáteční pozice shodovala s konečnou. Element <delay_start> určuje časový interval, který uplyne před spuštěním animace, a tento interval se opakuje před každým novým cyklem. Pokud je <auto_start> nastaven na *true*, animace se automaticky spustí ihned po inicializaci. Nastavuje se na *false* pouze v případě aktivace animace pluginem. [13] Příklad definice trajektorie jako sekvence pozic modelu je následující:

```
<trajectory id="0" type="walk">
      <waypoint>
        <time>0</time>
        <pose>10 0.4 1.0 0 0 0</pose>
      </waypoint>
      <waypoint>
        <time>2</time>
        <pose>13 0.4 1.0 0 0 0</pose>
      </waypoint>
      <waypoint>
        <time>2.5</time>
        <pose>13 0.4 1.0 0 0 1.57</pose>
      </waypoint>
```

```
      <waypoint>
        <time>4</time>
        <pose>13 2.4 1.0 0 0 1.57</pose
      </waypoint>
      <waypoint>
        <time>4.5</time>
        <pose>13 2.4 1.0 0 0 3.142</pose>
      </waypoint>
      <waypoint>
        <time>6</time>
        <pose>10 2.4 1.0 0 0 3.142</pose>
      </waypoint>
      <waypoint>
        <time>6.5</time>
        <pose>10 2.4 1.0 0 0 -1.57</pose>
      </waypoint>
      <waypoint>
        <time>8</time>
        <pose>10 0.4 1.0 0 0 -1.57</pose>
      </waypoint>
      <waypoint>
        <time>8.5</time>
        <pose>10 0.4 1.0 0 0 0</pose>
      </waypoint>
    </trajectory>
```

Element <time> udává časový interval, ve kterém má být daná poloha dosažena a element <pose> udává pozici a orientaci objektu v prostoru. [13] Ve výsledku model člověka následuje trajektorii „obdélníku“ pořád dokola.

4.2 Sledování RTF a výkonu CPU

V rámci projektu byly po konzultaci s vedoucím práce vytvořeny různé kombinace světů v Gazebo, které simulovaly rozdílné podmínky. Byly definovány dvě hlavní kategorie: svět se střechou (roof_world) a svět bez střechy (no_roof_world). Každá kategorie obsahuje dvě podkategorie založené na nastavení osvětlení – svět se všemi aktivovanými světly a svět, kde jsou světla aktivována pouze ve třech hlavních místnostech a na chodbě. Dále existují varianty s dveřmi otevřenými u všech místností a pouze u tří hlavních místností. Byla přidána také verze s dynamicky modelovaným člověkem.

Během testování těchto kombinací bylo monitorováno využití CPU, počet snímků za vteřinu (FPS) a hodnota RTF (Real Time Factor).

4.2.1 Testování světů

Nastavení otevřených či zavřených dveří, jako statické objekty, nemá vliv na výkon simulace. Stejně tak i svět se střechou a bez střechy.

Tabulka 4.1 Výsledky testování světů bez střechy a s otevřenými dveřmi

Název světa	RTF [%]	FPS	Využití CPU [%]
lights_ON_all_doors_open_world	99,1	40	29
lights_OFF_all_doors_open_world	99	59,9	25
actor_lights_ON_all_doors_open_world	98,8	38	30

Hlavní rozdíly byly pozorovány při změně nastavení osvětlení. Ačkoli RTF faktor nebyl významně ovlivněn, vyšší využití CPU v osvětleném světě vedlo k nižšímu počtu snímků za vteřinu, což způsobilo méně plynulý pohyb při změnách úhlů pohledu kamery. Ve světě s dynamickým modelem člověka bylo zaznamenáno nejvyšší využití CPU, což vedlo k nejpomalejší simulaci. Vložení jakéhokoliv robota dramaticky snižuje všechny klíčové metriky výkonu simulace, přičemž výkon se dále snižuje s počtem sensorů a kamer, které robot využívá. Výběr světa tak hraje důležitou roli pro plynulost a efektivitu simulace.

4.3 Optimalizace simulace

Provozování robotů v pomalém simulačním prostředí má řadu nevýhod, které negativně ovlivňují celkovou efektivitu a výkonnost. Mezi tyto problémy patří nesprávná funkce nebo nestabilní výsledky některých časově citlivých řídicích algoritmů, které nejsou schopny pracovat korektně při nízkém reálném časovém faktoru. Dále dochází k obtížím při škálování na více robotů, například v případě kooperaci robotů, nebo při přidávání složitějších prvků prostředí. Vývoj na méně výkonných laptopech se stává náročným, což omezuje možnosti práce v terénu. Nízká snímková frekvence vede k méně vizuálně atraktivním simulacím, což může být nevýhodou při prezentacích a vizualizacích výsledků. [14]

4.3.1 Snižování detailů 3D modelů

Každý model má svůj definovaný počet polygonů, který ovlivňuje jeho vizuální kvalitu a výpočetní náročnost. Modely s nízkým počtem polygonů, označované jako low-poly, jsou méně náročné na výpočetní výkon a nevyžadují výkonnou hardwarovou konfiguraci. Naproti tomu modely s vysokým počtem polygonů, známé jako high-poly, poskytují vyšší vizuální kvalitu, ale vyžadují výkonnější počítačové systémy. [15]. Pro simulace je proto vhodnější používat low-poly modely, protože složitější modely vyžadují více výpočetního času pro zpracování potenciálních kolizí a vykreslování.

V softwaru Blender lze zobrazit složitost modelu podle počtu polygonů. U modelů s velkým množstvím "trojúhelníků" je možné tyto polygony převést na "obdélníky", čímž se sníží celkový počet polygonů. Tato funkce, známá jako „*Tris to Quads“*, se nachází v kartě *Face* a byla využita například u modelu zdí.

Další metodou redukce počtu polygonů je použití nástroje *Decimate Modifier*. Tento nástroj efektivně snižuje počet vrcholů a ploch sítě s minimálními změnami tvaru, což z něj dělá ideální volbu pro složitější modely. Jednou z nevýhod je, že na rozdíl od většiny ostatních modifikátorů, *Decimate Modifier* neumožňuje vizualizovat změny v režimu úprav (Edit mode).[16]

4.3.2 Snížení počtu zdrojů světla a deaktivace stínů

Použití velkého počtu světelných zdrojů v simulaci může snížit Real Time Factor (RTF) v Gazebo. V případech, kdy je kladen důraz na přesnost simulace vzhledem k RTF, je vhodné deaktivovat některé přebytečné světelné zdroje. V reakci na to byly vytvořeny speciální verze světů, kde je osvětlení omezeno pouze na důležité místnosti.

Deaktivace stínů je dalším efektivním způsobem, jak zvýšit frekvenci snímků (FPS). Pro vyřazení stínů ve světě se používá následující XML kód:

```
<scene>
<shadows>0</shadows>
<grid>false</grid>
</scene>
```

Také deaktivování vestavěné mřížky může přispět ke zvýšení plynulosti simulace, i když tento efekt není tak výrazný.

4.3.3 Vhodné nastavení kamer a senzorů robota

Nastavení parametrů kamer a senzorů má zásadní vliv na plynulost simulace. Pro kamery a senzory typu pointcloud je vhodné snížit frekvenci aktualizace, omezit horizontální a vertikální zorné pole a zmenšit rozlišení výstupního obrazu (platí pouze pro kamery). U LiDAR senzorů je doporučeno snížit maximální dosah, frekvenci aktualizace a upravit úhlové rozlišení, ideálně na 1°. [14]

5. TESTOVÁNÍ ROBOTA

Funkčnost vytvořených světů je ověřena vložením robota vybaveného senzorem LiDAR a kamerou. Robot je definován pomocí XML kódu a je integrován do simulovaného světa.

5.1 Senzor LiDAR

Senzor LiDAR (Light Detection and Ranging), funguje na principu vysílání infračervených laserových paprsků, které se odrážejí od objektů v prostředí a vracejí zpět k přijímači. Na základě časového intervalu mezi vysláním paprsku a jeho přijetím se určuje vzdálenost překážek, od kterých se paprsek odrazil. [17] LiDAR senzory mohou fungovat za různých světelných podmínek, včetně tmy.

5.2 Vytvoření modelu robota

Model robota je psaný v XML kódu jako vytvořený svět pro jednoduché vložení a rychlé otestování funkčnosti. Jelikož je robot dynamický model na následujících řádcích je specifikován postup vytvoření robota pro Gazebo.

5.2.1 Definice robota

Příklad kódu:

```
<model name='vehicle_blue' canonical_link='chassis'>
    <pose relative_to='world'>12 7 0.5 0 0 0</pose>
```

Nejdříve je definováno jméno modelu, které by mělo být jedinečný mezi ostatními tagy, nebo modely na stejné úrovni. Každý model může mít jeden odkaz to určuje *canonical_link,* tento odkaz je poté připojen k implicitnímu rámci modelu, v tomto případě se se to vztahuje na *chassis*. Pokud kanonický odkaz není definován, tak bude vybrán první <link> a k němu přiřazen. Element <pose> určuje polohu a orientaci modelu. Atribut *relative_to* definuje pozici modelu vůči světu. Tedy po spuštění simulace bude model začínat na dané pozici ve světě. [18]

5.2.2 Podvozek robota

Každý model robota v Gazebo je sestaven z několika segmentů, označovaných jako *link*, které jsou vzájemně propojeny pomocí kloubů neboli *joint*. Následující kód ukazuje definici podvozku (chassis) robota a jeho pozici relativní k celému modelu: [18]

```
  <link name='chassis'>
        <pose relative_to='__model__'>0.25 0 0.2 0 0 0</pose>
```

5.2.3 Inerciální vlastnosti

Inerciální vlastnosti jsou dále specifikovány jako hmotnost modelu a matice momentu setrvačnosti. Vzorec výpočtu setrvačnosti obdélníkové desky kolem jejího těžiště se počítá kolem os x,y,z. Vzorec pro moment setrvačnosti kolem osy x je

$$Ix = \frac{M}{12} \cdot a^2 \,, \tag{5.1}$$

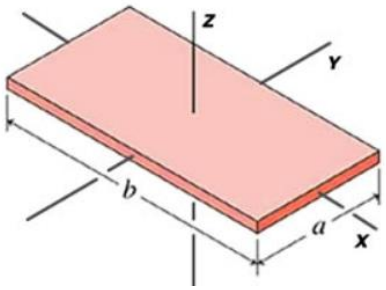
kde *M* je hmotnost kvádru a parametr *a* je šířka obdélníkové desky. Vzorec pro moment setrvačnosti kolem osy y je

$$Iy = \frac{M}{12} \cdot b^2 \,, \tag{5.2}$$

kde *M* je hmotnost kvádru a parametr *b* je délka obdélníkové desky. Vzorec pro moment setrvačnosti kolem osy z je

$$Iz = \frac{M}{12} \cdot (a^2 + b^2) \,, \tag{5.3}$$

kde *M* je hmotnost kvádru a parametr *a* je šířka *b* je délka obdélníkové desky.



Obrázek 5.1 Obrázek kvádru z označenými osami [19]

Matici momentu setrvačnosti se vypočítala použitím nástroje, které Gazebo doporučuje [19]. Hmotnost šasi robota je zvolena na 0,5 kg. Výsledná matice momentu setrvačnosti je

$$Ir = \begin{pmatrix} 0.010417 & 0 & 0 \\ 0 & 0.041667 & 0 \\ 0 & 0 & 0.052083 \end{pmatrix}. \tag{5.4}$$

V následujícím kódu jsou definovány vypočtený moment setrvačnosti a hmotnost šasi robota:

```
<inertial>
            <mass>0.5</mass>
```

```
<inertia>
                <ixx>0.010417</ixx>
                <ixy>0</ixy>
                <ixz>0</ixz>
                <iyy>0.041667</iyy>
                <iyz>0</iyz>
                <izz>0.052083</izz>
            </inertia>
</inertial>
```

Element <mass> označuje hmotnost šasi robota, zatímco <inertia> specifikuje jednotlivé složky matice momentu setrvačnosti. [18]

5.2.4 Vizualizace a kolize

Vizualizace modelu je definována specifikací tvaru a barevných vlastností. Příklad kódu pro vizuální reprezentaci modelu:

```
<visual name='visual'>
            <geometry>
                <box>
                    <size>1.0 0.5 0.25</size>
                </box>
            </geometry>
            <material>
                <ambient>0.0 0.0 1.0 1</ambient>
                <diffuse>0.0 0.0 1.0 1</diffuse>
                <specular>0.0 0.0 1.0 1</specular>
            </material>
</visual>
```

Element <geometry> obsahuje tvar modelu, zde definovaný jako kvádr <box> s určenými rozměry. Element <material> specifikuje materiálové vlastnosti jako ambientní, difuzní a spekulární barvy, které jsou v tomto případě nastaveny na modrou.

Kolize modelu je definována tím, že určuje tvar, který se použije při srážkách modelu s ostatními objekty. Příklad kódu pro kolizní model: [18]

```
<collision name='collision'>
            <geometry>
                <box>
                    <size>1.0 0.5 0.25</size>
                </box>
            </geometry>
</collision>
```

Kolizní tvar může být stejný nebo odlišný od vizuálního tvaru. Často se používají jednodušší geometrické tvary pro kolizní modely k snížení výpočetní zátěže a zrychlení simulace.

5.2.5 LiDAR senzor

Konfigurace LiDAR senzoru začíná vytvořením rámu (frame), do kterého je senzor upevněn. Tento rám je připojen k podvozku robota, jak demonstruje následující kód:

```
<frame name="lidar_frame" attached_to='chassis'>
        <pose>0.4 0 0.25 0 0 0</pose>
</frame>
```

Pro integraci senzoru do simulace je dále potřeba přidat plugin pro senzory. Následující XML kód definuje potřebný plugin:

```
<plugin
      filename="gz-sim-sensors-system"
      name="gz::sim::systems::Sensors">
      <render_engine>ogre2</render_engine>
</plugin>
```

Následně je specifikován samotný senzor LiDAR s důležitými parametry pro funkčnost simulace a správné zobrazení dat:

```
<sensor name='gpu_lidar' type='gpu_lidar'>
            <pose relative_to='lidar_frame'>0 0 0 0 0 0</pose>
            <topic>lidar</topic>
            <update_rate>10</update_rate>
          <lidar>
            <scan>
              <horizontal>
                <samples>640</samples>
                <resolution>1</resolution>
                <min_angle>-1.396263</min_angle>
                <max_angle>1.396263</max_angle>
              </horizontal>
              <vertical>
                <samples>1</samples>
                <resolution>1</resolution>
                <min_angle>0.0</min_angle>
                <max_angle>0.0</max_angle>
              </vertical>
            </scan>
            <range>
              <min>0.08</min>
              <max>10.0</max>
              <resolution>0.01</resolution>
            </range>
          </lidar>
        </sensor>
```

Jméno a typ senzoru jsou definovány na začátku. Element <topic> je klíčový pro komunikaci mezi Gazebo a ROS2, jelikož určuje, na jaký topic budou data z LiDARu publikována. Frekvence aktualizace dat je definována v <update_rate>. Vlastnosti horizontálních a vertikálních paprsků jsou specifikovány pod <horizontal> a <vertical>.

U elementu <samples> se definuje počet simulovaných lidarových paprsků, které se generují při každém kompletním cyklu laserového skenu. Číslo pod elementem <resolution> se vynásobí vzorky (samples) a udává počet bodů datového rozsahu. Elementy <min_angle> a <max_angle> jsou úhlové rozsahy generovaných paprsků. Rozsah každého jednoho paprsku definuje element <range>. Subelementy <min> a <max> definují minimální a maximální vzdálenost pro každý LiDAR paprsek. Lineární rozlišení každého paprsku LiDARu je určeno subelementem <resolution>. Element <always_on> pokud je *true* senzor bude aktualizován podle <update_rate>. Nakonec element <visualize> pokud je *true*, senzor bude vizualizován v grafickém uživatelském rozhraní (GUI). [20]

5.2.6 Kamera

Konfigurace kamery v simulaci robota je podobně jako u LiDARu implementována pomocí specifikovaného XML kódu, který umožňuje definovat její vlastnosti a funkce. Příklad kódu pro kamerový senzor:

```
<sensor name="camera1" type="camera">
        <camera name="camera_front">
            <horizontal_fov>1.4137</horizontal_fov>
            <lens>
                <intrinsics>
                    <fx>1108.952913</fx>
                    <fy>1110.658360</fy>
                    <cx>729.53</cx>
                    <cy>544.98</cy>
                    <s>1</s>
                </intrinsics>
            </lens>
            <distortion>
                <k1>0.0</k1>
                <k2>0.0</k2>
                <k3>0.0</k3>
                <p1>0.0</p1>
                <p2>0.0</p2>
                <center>0.5 0.5</center>
            </distortion>
            <image>
                <width>1440</width>
                <height>1080</height>
                <format>R8G8B8</format>
            </image>
            <noise>
                <type>gaussian</type>
                <mean>0</mean>
                <stddev>0.007</stddev>
            </noise>
        </camera>
      </sensor>
  </link>
```

Konfigurace objektivu (lens) zahrnuje parametry, jako jsou ohniskové vzdálenosti (fx, fy), střed obrazu (cx, cy) a měřítko zkreslení (s). Kromě toho je specifikováno zkreslení objektivu a parametry pro vytváření obrazu, jako jsou rozměry obrazu a formát

Vizuální šum je modelován jako Gaussovo rozložení s definovanými parametry pro střední hodnotu a standardní odchylku, což simuluje realistický obrazový šum v reálném světě.

5.2.7 Kola podvozku robota

Stejně jako u modelu podvozku robota, vytvoříme model kola a připojíme ho k boku podvozku. Vzhledem k tomu, že kolo má tvar válce, klíčovou změnou v kódu je úprava geometrie, jak je ilustrováno v následujícím XML kódu:

```
<geometry>
                <cylinder>
                    <radius>0.2</radius>
                    <length>0.1</length>
                </cylinder>
</geometry>
```

V geometrii válce je definován rádius (radius) a délka (length). Druhé kolo je konfigurováno obdobně a připojeno na opačný bok podvozku robota. Tímto způsobem je zajištěna symetrie a funkčnost pohybového mechanismu robota.

Pojezdové kolo má tvar koule, je tedy oproti kolu, které je na boku podvozku, změněna geometrie. Tedy místo válce (cylinder) je definována koule neboli sféra (sphere) s daným rádiusem.

5.2.8 Propojení částí modelu

Propojování částí robota se provádí pomocí kloubů (joints), které zajišťují vzájemné spojení segmentů (links) a definují, jak budou tyto části vzájemně interagovat. [18] Příklad kódu pro připojení levého kola k podvozku:

```
<joint name='left_wheel_joint' type='revolute'>
    <pose relative_to='left_wheel'/>
    <parent>chassis</parent>
    <child>left_wheel</child>
<axis>
      <xyz expressed_in='__model__'>0 1 0</xyz>
             <limit>
                  <lower>-1.79769e+308</lower>
                  <upper>1.79769e+308</upper>
             </limit>
      </axis>
</joint>
```

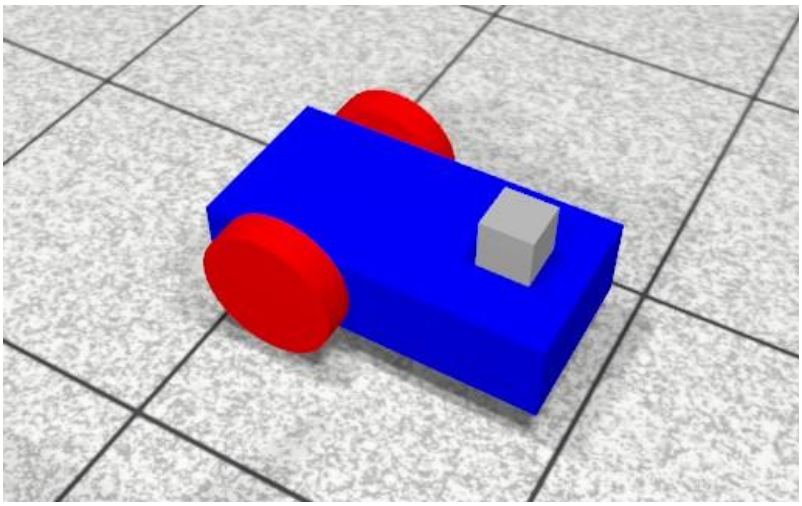
V tomto příkladu je kloub pojmenován jako <joint> a je definován jeho typ *revolute*, který poskytuje jeden rotační stupeň volnosti. Pozice kloubu je určena relativně k levému kolu, a osa rotace je definována podél osy y. Pro kloub typu *revolute* je klíčové definovat limity rotačního úhlu (<limit>), zde nastavené na teoretické maximum a minimum. [18]

Pro pojezdové kolo je nastaven typ kloubu na *ball*, což robotu umožní tři rotační stupně volnosti. [18]

5.2.9 Ovládání pohybu robota

Ovládání robota je přes klávesnici. K této funkčnosti jsou vloženy dva pluginy. První je *Key Publisher*, který je součástí GUI Gazeba dokáže číst stisky kláves a odesílat je na výchozí topic /keyboard/keypress. Druhý plugin *Triggered Publisher* tyto stisky kláves mapuje do zpráv typu *Twist* a publikuje je na topic /cmd_vel, který model robota poslouchá. [21] Zde je příklad kódu, který mapuje šipku nahoru, jako pohyb vpřed:

```
<!-- Moving Forward-->
        <plugin filename="gz-sim-triggered-publisher-system"
                name="gz::sim::systems::TriggeredPublisher">
            <input type="gz.msgs.Int32" topic="/keyboard/keypress">
                <match field="data">16777235</match>
            </input>
            <output type="gz.msgs.Twist" topic="/cmd_vel">
                linear: {x: 0.5}, angular: {z: 0.0}
            </output>
</plugin>
```



Obrázek 5.2 Model robota simulovaný v Gazebu

5.3 Vizualizace dat senzoru LiDAR a kamery v ROS2

Aby bylo možné komunikovat se simulací Gazebo pomocí ROS2, je nezbytné použít balík zvaný ros_ign_bridge. Tento balík slouží jako síťový most, který přeposílá zprávy mezi ROS2 a Gazebo, umožňující obousměrnou komunikaci mezi oběma systémy. [22]

5.3.1 Vytvoření síťového mostu

Pro efektivní přenos dat z LiDARu je použit topic /lidar, který je v ROS2 přístupný jako /laser_scan. Následuje příklad příkazu pro vytvoření mostu v terminálu, který propojuje tyto dva topicy:

```
ros2 run ros_gz_bridge parameter_bridge /lidar@sensor_msgs/msg/LaserScan[ignition.msgs.LaserScan --ros-args -r /lidar:=/laser_scan
```

Podobně se vytváří most pro data z kamery, která jsou dostupná na topicu /camera a v ROS2 jsou přístupná jako /camera_image. Zde je příklad vytvoření mostu pro přenos obrazových dat:

```
ros2 run ros_gz_bridge parameter_bridge /camera@sensor_msgs/msg/Image[ignition.msgs.Image --ros-args -r /camera:=/camera_image
```

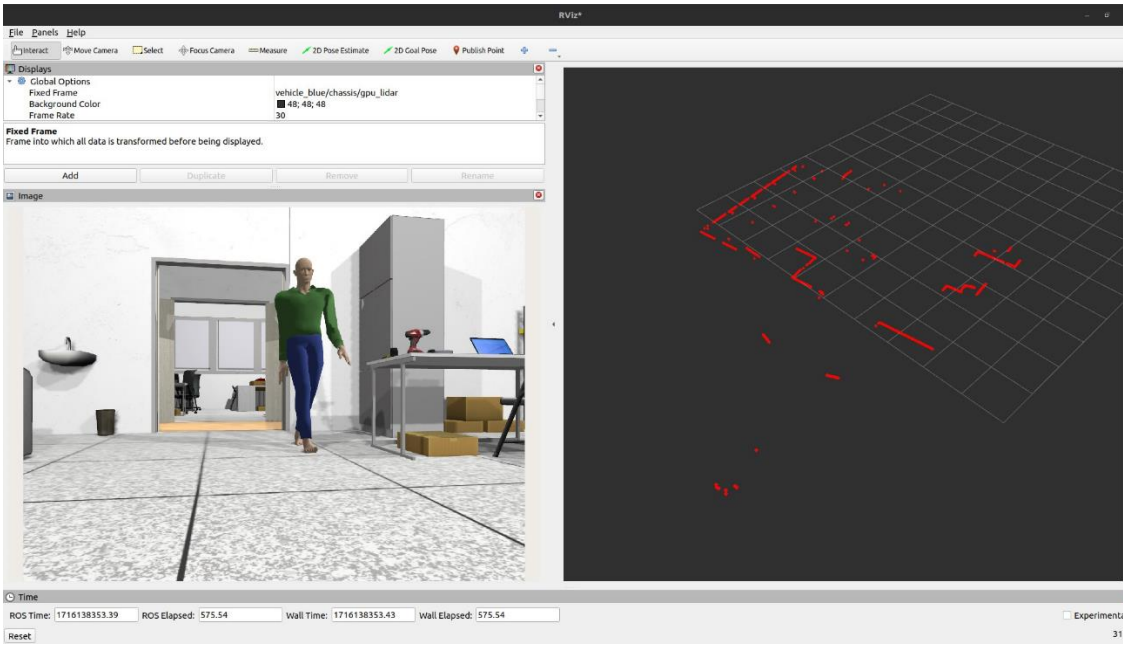
5.3.2 Rviz

Pro efektivní použití Rviz je nezbytné správně nastavit *Fixed Frame*. Tento základní referenční rámec slouží jako výchozí bod, k němuž jsou vztahovány všechny další rámy vytvořené během modelování robota. Správná konfigurace *Fixed Frame* je klíčová pro korektní zobrazení a interpretaci prostorových dat v 3D prostředí. Níže je příklad, jak nastavit *Fixed Frame* v Rviz pro senzor LiDAR:

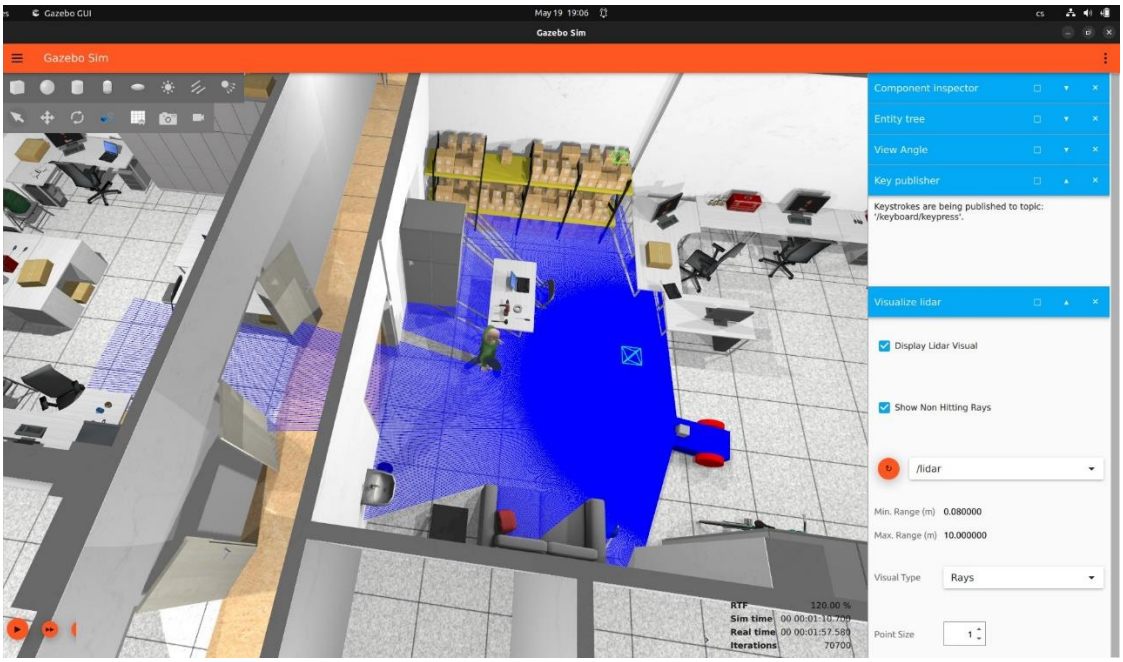
```
vehicle_blue/chassis/gpu_lidar
```

V tomto příkladu, vehicle_blue je název modelu robota, chassis je rám, ke kterému je senzor připojen, a gpu_lidar je jméno samotného senzoru. Toto nastavení zajišťuje, že data zobrazená v Rviz budou správně orientována vzhledem k modelu robota a jeho struktuře.

Přidáním záložky *LaserScan* se po spuštění Gazebo simulace začnou zobrazovat data z LiDARu. Podobným způsobem je přidána záložka *Image*, která zobrazí data z kamery.



Obrázek 5.3 Výsledná vizualizace dat ze senzoru LiDARu (vpravo) a kamery (vlevo)



Obrázek 5.4 Referenční stav simulace v Gazebu v porovnání s Rviz. Použit byl svět s vloženým robotem, bez střechy, s dynamickým modelem člověka a se všemi rozsvícenými zdroji světla.

6. ZÁVĚR

Tato práce se zaměřovala na seznámení se s frameworkem ROS2 a simulačním prostředím Gazebo v systému Ubuntu. Dále zahrnovala seznámení s tvorbou světů a vytvořením modelu. Prvním krokem bylo stažení a instalace ROS2 a Gazebo. Po studiu dokumentace a vypracování tutoriálů k ROS2 a Gazebo byl splněn první bod zadání.

Další část práce zahrnovala vytvoření modelu kostky, přičemž každá plocha byla opatřena texturou podle pokynů vedoucího práce. Byly popsány postupy pro tvorbu vlastního modelu a výsledný model byl simulován ve vytvořeném světě Gazebo čímž byl splněn druhý bod zadání.

Poté byl vytvořen model chodby a místností, doplněný o modely získané z oficiálních stránek Gazebo a modely vytvořené v Solidworks. Byly vytvořeny různé kombinace světů s odlišnými podmínkami, dle pokynů vedoucího. Vytvořené modely a světy byli ukládány do volně přístupného GitHub repozitáře s příslušnou dokumentací umožňující snadné stažení a spuštění. Odkaz na repozitář je k dispozici zde: https://github.com/josefchtr/workplace-VUT-FEKT-UAMT-world.git. Tento repozitář je i formou přílohy k práci, jelikož příloha má příliš velkou kapacitu, tak se nemohla vejít do IS (informačního systému), je tedy umístěna pouze na datový nosič.

Následně byly vytvořené světy testovány sledováním hodnot RTF a využití CPU. Největší vliv na plynulost simulace měly světy s aktivovanými všemi zdroji světla a dynamickým modelem člověka. Pro zajištění rychlejšího chodu simulace mělo největší vliv deaktivování stínů. U modelů měl na plynulost simulace vliv počet polygonů modelu. U některých modelů byl snížen počet polygonů pomocí různých metod. Nejrychlejší metodou byla *Tris to Quads*.

Ve finální fázi byla otestována funkčnost simulace světa vložením robota osazeného senzorem LiDAR a kamerou. Tento výsledný svět Gazeba byl propojen s ROS2 a vizualizován v nástroji Rviz. Výsledky naznačují, že vytvořené světy mohou být účinně využity v praxi pro testování robotů.

7. LITERATURA

[1] *Ros 2 Documentation: Humble*, 2022. Online. Aktualizováno 17.4.2024. Dostupné z: https://docs.ros.org/en/humble/index.html. [cit. 2023-12-19].

[2] *About Gazebo*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/about. [cit. 2023-12-19].

[3] *SolidWorks*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, [2009], 23. 5. 2023. Dostupné z: https://cs.wikipedia.org/wiki/SolidWorks. [cit. 2023-12-19].

[4] *About blender*, [2002]. Online. Blender. Dostupné z: https://www.blender.org/about/. [cit. 2023-12-19].

[5] *SDFormat*. Online. SDFormat Home. 2020. Dostupné z: http://sdformat.org/. [cit. 2023-12-20].

[6] *SDF Worlds*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/docs/garden/sdf_worlds#sdf-worlds. [cit. 2024-05-16].

[7] *Soubory STL*, [2023]. Online. Adobe. Dostupné z: https://www.adobe.com/cz/creativecloud/file-types/image/vector/stl-file.html. [cit. 2024-05-10].

[8] *Make a model*, 2014. Online. Gazebo Classic. Dostupné z: https://classic.gazebosim.org/tutorials?tut=build_model. [cit. 2024-05-10].

[9] *Model structure and requirements*. Online. GazeboSim. 2014. Dostupné z: https://classic.gazebosim.org/tutorials?tut=model_structure. [cit. 2023-12-21].

[10] *Models*, [2022]. Online. Gazebo Sim. Dostupné z: https://app.gazebosim.org/fuel/models. [cit. 2024-05-12].

[11] *Free 3D Models*, [2013]. Online. Dostupné z: https://free3d.com/. [cit. 2024-05-12].

[12] *Beginner: GUI*, [2014]. Online. Gazebo. Dostupné z: https://classic.gazebosim.org/tutorials?tut=guided_b2. [cit. 2024-05-16].

[13] *Actors*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/docs/garden/actors. [cit. 2024-05-16].

[14] *5 Ways to Speedup Gazebo Simulations,* [2023]. Online. Black coffee robotics. Dostupné z: https://www.blackcoffeerobotics.com/blog/5-ways-to-speedup-gazebo-simulations. [cit. 2024-05-17].

[15] BEDNÁŘ, Miroslav; KRÁKORA, David a ŠIMON, Michal, 2021. *Optimalizace 3D modelů pro virtuální realitu*. Online, konferenční příspěvek. Plzeň: Západočeská univerzita v Plzni. ISBN 978-80-261-0792-7. Dostupné z: https://dspace5.zcu.cz/bitstream/11025/46397/1/PI2021%20-%20sbornik_komplet-19-27.pdf. [cit. 2024-05-17].

[16] *Decimate Modifier*, [2023]. Online. Blender. 23.04.2024. Dostupné z: https://docs.blender.org/manual/en/latest/modeling/modifiers/generate/decimate.html. [cit. 2024-05-17].

[17] CIHLÁŘ, Miloš, 2020. *Srovnáni 2D LIDAR SLAM metod v simulaci a v reálném světě*. Bakalářská práce. Technická 3082/12 616 00 Brno Česká republika: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií.

[18] *Building your own robot*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/docs/garden/building_robot. [cit. 2024-05-19].

[19] *Amesweb - Advanced Mechanical Engineering Solutions*, c2013-2024. Online. Dostupné z: https://amesweb.info/inertia/mass-moment-of-inertia-calculator.aspx. [cit. 2024-05-19].

[20] *Sensors*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/docs/garden/sensors. [cit. 2024-05-19].

[21] *Moving The Robot*, [2022]. Online. Gazebo. Dostupné z: https://gazebosim.org/docs/garden/moving_robot. [cit. 2024-05-19].

[22] *Setting up a robot simulation (Gazebo)*, 2023. Online. Ros 2 Documentation. 31.05.2023, 21.02.2024. Dostupné z: https://docs.ros.org/en/foxy/Tutorials/Advanced/Simulators/Ignition/Ignition.html. [cit. 2024-05-19].

SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

FEKT	Fakulta elektrotechniky a komunikačních technologií
VUT	Vysoké učení technické v Brně
2D	Dvojrozměrný
3D	Trojrozměrný
UAMT	Ústav automatizace a měřící techniky
SDF	Simulation Description Format
XML	Extensible Markup Language
STL	Standard Triangle Language format
DAE	Digital Asset Exchange file
RTF	Real Time Factor
CPU	Central Processing Unit
LiDAR	Light Detection And Ranging
Rviz	vizualizační nástroj v prostředí ROS2

Symboly:

m	hmotnost	(kg)