VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta strojního inženýrství

DIPLOMOVÁ PRÁCE

Brno, 2016 Bc. Dušan Borek

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

SIMULACE DYNAMIKY VOZIDLA VYUŽITÍM SOFTWARE V-REP

SIMULATION OF VEHICLE DYNAMICS USING V-REP SOFTWARE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Dušan Borek**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Robert Grepl, Ph.D.**
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Bc. Dušan Borek**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Mechatronika
Vedoucí práce:	**doc. Ing. Robert Grepl, Ph.D.**
Akademický rok:	2015/16

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Simulace dynamiky vozidla využitím software V-REP

Stručná charakteristika problematiky úkolu:

Nejznámnější ukázku propracované jízdní simulace vozidel můžeme vidět v současných počítačových hrách. Dokonalá grafika je doplněna i velmi realistickým modelem dynamiky vozu a jeho kontaktu s terénem.
V této práci se budeme zabývat podobnou tématikou. S vužitím software pro robotiku a vizulizaci vytvořit simulaci reálného modelu čtyřkolového vozidla a následně testovat různé možnosti, které tento software nabízí.

Cíle diplomové práce:

1. Představení software V-REP v kontextu modelování, robotiky a vizualizace. Stručný popis principu fungování a možností.
2. Úprava modelu vozidla projektu Car4 a následná implementace ve V-REP. Testování několika variant modelu vozidla (různá složitost dynamiky vozidla a vizualizovaných detailů).
3. Ovládání modelu pomocí vestavěného programovacího rozhraní ve V-REP a pomocí externí aplikace (MATLAB).
4. Experimentování s možnostmi software V-REP. Testování možností simulace sensorických informací (sensory vzdálenosti), porovnání řešičů ODE, vliv na rychlost simulace.

Seznam literatury:

Vlk, F.: Dynamika motorových vozidel. 2nd edition. Brno: František Vlk, 2003.434 p. ISBN 80-23--0024-2.

Noskievič P.: Modelování a identifikace systémů, 1999

Valášek, M.: Mechatronika, Vydavatelství ČVUT 1995

ABSTRAKT

Tato práce se zabývá modelováním dynamiky a vizualizací experimentálního vozidla CAR4 v programu V-REP. V první části se práce zabývá představením a možnostmi programu V-REP. V další části je realizováno modelování vozidla CAR4 s následným řízením. K řízení je využit program V-REP a Matlab. V poslední fázi proběhlo testování modelu v simulaci.

KLÍČOVÁ SLOVA

simulace, dynamika, vizualizace, CAR4, model, modelování, auto, V-REP, Matlab, 4WS

ABSTRACT

This theasis deals with modeling dynamics and visualization of experimental vehicle CAR4 in software V-REP. The first part shows introduction with V-REP and its possibilities for usage in simulations. The second part presents modelling of CAR4 vehicle and its control by using software V-REP and Matlab. In last part are models of vehicle tested in simulation)

KEYWORDS

simulation, dynamics, visualization, CAR4, model, modeling, car, V-REP, Matlab, 4WS

BOREK, Dušan *Simulace dynamiky vozidla využitím software V-REP*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mechaniky těles, mechatroniky a biomechaniky, 2016. 57 s. Vedoucí práce byl doc. Ing. Robert Grepl, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Simulace dynamiky vozidla využitím software V-REP“ jsem vypracoval(a) samostatně pod vedením vedoucího diplomové 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(ka) uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu doc. Ing. Robertu Greplovi, Ph.D. za zprostředkování zajímavého tématu diplomové práce. Dále bych poděkoval své rodině za duševní a finanční podporu během vysokoškolského studia.

Brno ....

podpis autora(-ky)

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Toto diplomové téma je zaměřena na práci se software V-REP. Jedná se o nástroj pro simulaci a vizualizaci modelů pro robotiku.

V dnešní technicky rozvinuté době již téměř není možné se setkat s řešením inženýrských problémů bez jejich zjednodušení výpočetní technikou. Nejčastěji se tak děje v podobě modelování pomocí technických výpočtů. Existuje již nepřeberné množství programů, v rámci mechatroniky např. hojně využívaný Matlab, které se těmito výpočty zabývají. Z pohledu simulace kinematiky a dynamiky v robotice je situace velmi obdobná, ale málokdy je možné získaná data ihned interpretovat v podobě realitě se blížící vizualizace. Spousta dynamických výpočtů je již také převedených do sofwarové podoby prostřednictvím open-source knihoven dostupnými na internetu.(např. Bullet Physics Library nebo Open Dynamics Engine). V jádru program V-REP nabízí tyto knihovny již zabudované spolu s modelačním a výpočtovým prostředím. Je tedy možné pouze namodelovat robotický problém spolu se stanovením parametrů a počátečních podmínek výpočtu. Výsledkem bude vizualizace problému založená na dynamických výpočtech. Je také možnost, díky zabudovanému skriptovacímu rozhraní, tyto simulace řídit. Případně také propojit s jinými programovacími jazyky jako C++, Python, Matlab, atd..

Díky těmto vlastnostem se V-REP zdá být poměrně schopným nástrojem pro vizualizaci a robotiku a jeho možnosti budou blíže prozkoumány v této diplomové práci.

1 FORMULACE PROBLÉMU A CÍLE ŘEŠENÍ

Pro testování možností software V-REP v oblasti simulace vozidla je důležitý výběr předlohy. K tomuto účelu se skvěle hodí experimentální vozilo CAR4, které vzniklo v roce 2010 v rámci několika diplomových a bakalářských prací a v dalších letech na něm byly následně prováděny úpravy mechaniky a elektroniky.

Projekt CAR4 je čtyřkolové vozidlo s nezávislých pohonem pro každé kolo. Pro každé kolo tak lze libovolně nastavovat úhel natočení kola, realizované servomotorem. Totéž platí pro rychlost kola, realizované otáčkami elektromotoru určeného pro pohon. Vzhledem k nezávislému pohonu všech kol má model odlišnou konstrukci uchycení kol, než je tomu běžné u tradičních automobilů.

Cíle práce:

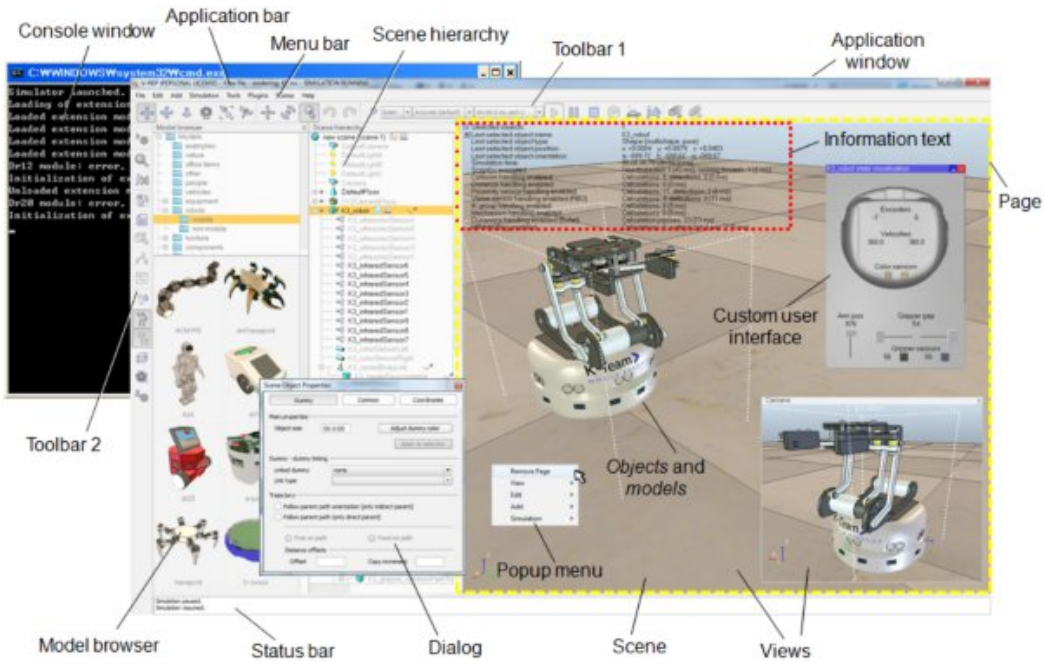
- Prozkoumání základních funkcí software V-REP z hlediska modelování, z hlediska řízení modelů užitím algoritmizace, možnostmi kooperace s jinými softwary a možnosti simulace senzorů
- Vytvoření modelu pro základní představu o postupu při modelování
- Vytvoření komplexního modelu se zachováním reálných vazeb experimentálního vozidla CAR4
- Řízení modelu využitím skriptovacích možností software V-REP
- Řízení modelu využitím software Matlab
- Prozkoumat možnosti virtuálních senzorů
- Prozkoumat možností real-time simulace a komunikace se software Matlab
- Nalezení a využití zajímavých funkcí pro model vozidla

2 PŘEDSTAVENÍ SOFTWARE V-REP A OBLAST VYUŽITÍ

2.1 Uživatelské rozhraní [2]

Software V-REP disponuje integrovaným vývojovým a programovatelným prostředím.2.1

Pro tento účel je také navržen uživatelský interface:



Obr. 2.1: Uživatelské rozhraní [2]

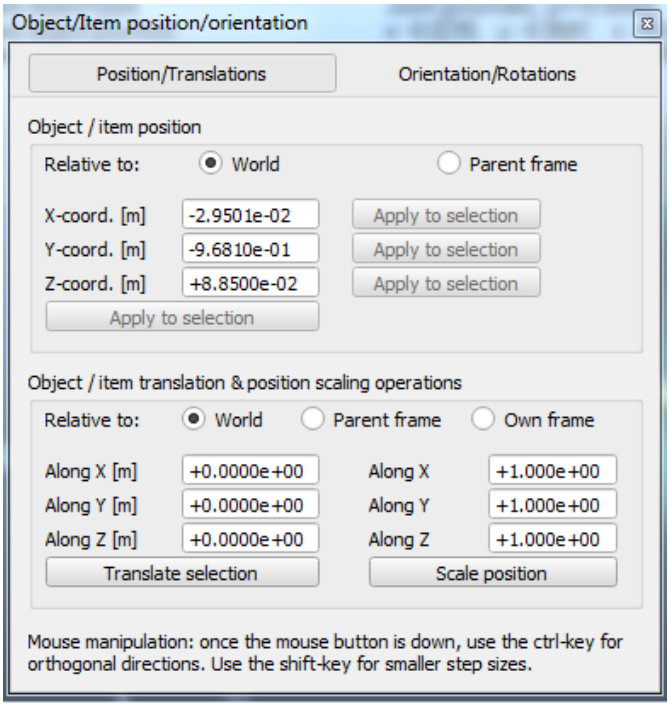
legenda:

- **console window:** Při běhu aplikace pod operačním systémem Windows je při spuštění vytvořeno příkazové okno. Ve výchozím nastavení je skryto.
- **application window:** Jedná se o hlavní okno aplikace.
- **several dialogs:** Pomocí těchto nabídek lze měnit konkrétní nastavení a parametry.
- **application bar:** Klasická popisová lišta programu.
- **menu bar:** Přístup k téměř všemu nastavení, které je k dispozici.
- **toolbars:** Tato lišta obsahuje často využívané funkce programu.
- **model browser:** Knihovna obsahující již vytvořené modely ve V-REP včetně příkladových modelů.

- **scene hierarchy:** Obsahuje stromovou hierarchii aktivní scény.
- **page:** Každá scéna může obsahovat až 8 stran, přičemž každá strana může obsahovat neomezené množství pohledů.
- **views:** Pohledy jsou používané k žádanému zobrazení scény.
- **information text:** Zobrazuje informace o aktivním objektu a nastavení simulace.
- **status bar:** Zobrazuje aktuální informace o vykonání příkazů a chybových hláškách.
- **custom user interfaces:** Okno s nastaveními vytvořené samotným uživatelem a přiřazené dané scéně.
- **popup menu:** Menu, které se zobrazí po kliknutí pravého tlačítka myši, obsahující většinu funkcí, které jsou právě k dispozici.

2.2 Manipulace s objekty [2]

S veškerými objekty ve scéně lze manipulovat pomocí nabídky Object Shift, a poté tažením kurzoru myši, nebo ideálně změnou parametrů v souřadnicovém systému. Toho lze docílit vzhledem k absolutnímu SS („Souřadnicový systém“ - Coordinate system), relativnímu SS nadřazeného objektu nebo relativnímu SS sebe samého.



Obr. 2.2: Rozhraní pro manipulaci s objekty

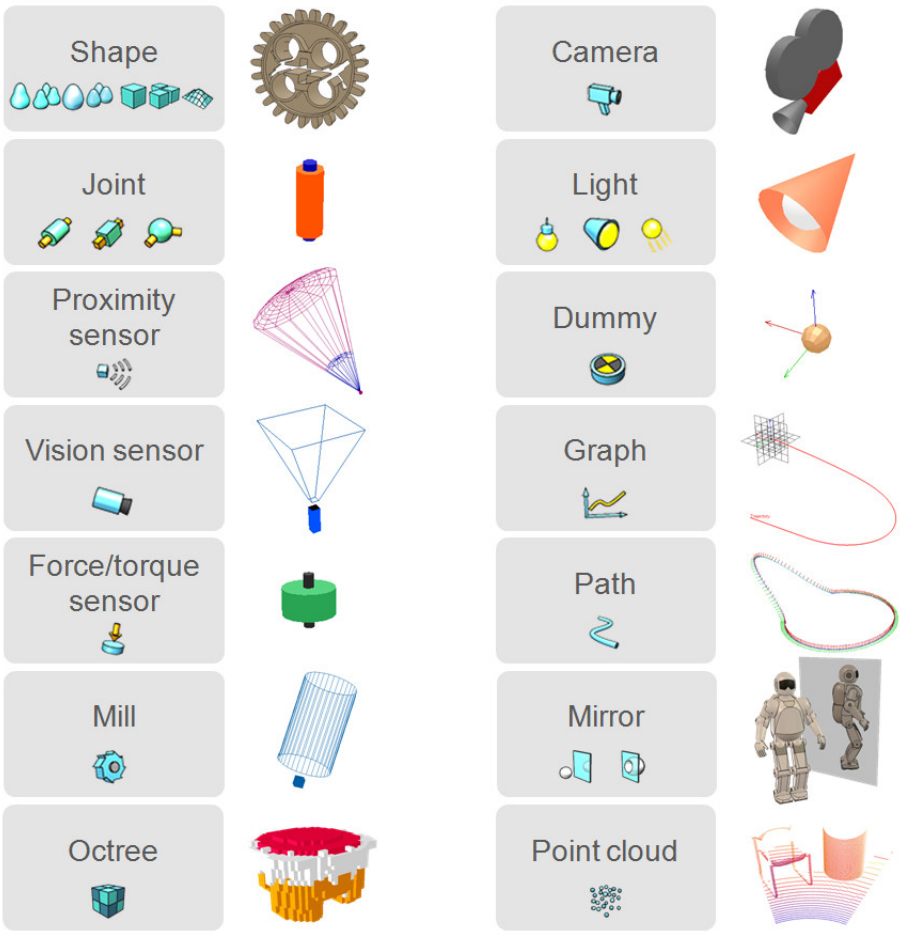
Je proto vhodné při větším množství objektů ve scéně využívat stromové struktury, a určovat mateřské objekty, vůči kterým následně umísťovat další objekty. Výhoda tohoto přístupu je, že při manipulaci s takovým mateřským objektem bude mít za následek změnu polohy také všech podřízených objektů. Obdobným způsobem lze s objekty také rotovat pomocí nabídky Object Rotate. Realizace probíhá zadáním žádaného úhlu rotace ve stupních kolem určité osy. Taktéž je možno rotovat pomocí tažení kurzoru myši. Jiný způsob manipulace s objekty není možný, což má za následek to, že modelování ve V-REP je poměrně zdlouhavý proces.

2.3 Scéna a objekty [2]

Scéna je hlavní pracovní plocha programu. Každá scéna je složena z modelů, případně jednotlivých objektů. Veškeré objekty, které je možné ve V-REP využít, jsou zobrazeny na obrázku 2.3

Mezi hlavní objekty patří:

- Tvary - Jednoduché tvary je možné vytvořit pomocí zabudovaného nástroje, nebo ty složitější je možné importovat např. ve formátu .dxf nebo .stl, atd. Pro efektivní spolupráci tvaru s dynamickým enginem je nutné tento tvar převést pomocí vestavěného nástroje do konvexního tvaru.
- Vazby - Jsou k dispozici vazby Revolute (rotační), prismatic (posuvná) a spherical (sférická).
- Kamery - Slouží pro zobrazení scény, objektů a detailů. Je možně jejich libovolné umístění.
- Osvětlení - Je možnosti upravit osvětlení scény.
- Grafy - V-REP nabízí široké možnosti zobrazení jednotlivých parametrů pomocí grafů. Od absolutních a relativních vzdálenosti a rychlostí objektů, přes parametry scény a simulace, úhlové rychlosti, atd
- Dummy - Dummy slouží k mnoha účelům, nejčastěji k propojování objektů s vazbami ve stromové struktuře scény nebo jako vhodné programovatelné objekty.
- Proximity senzory - Je možné používat pro zjišťování vzdáleností v reálném čase mezi objekty.
- Vizuální senzory - Umožňují získat obrazovou matici ze scény pro další zpracování.
- Silové senzory - Slouží ke spojování statických objektů a také k měření silové interakce mezi objekty.



Obr. 2.3: Všechny typy objektů, které je možné využít ve V-REP [2]

U všech typů objektů je také možnost nastavit vlastnosti v průběhu simulace:

- Collidable - toto nastavení určuje, zda objekt může či nemůže přijít do kolize s jiným objektem
- Measurable - toto nastavení určuje, že mezi více měřitelnými objekty může být definována minimální vzdálenost
- Detectable - zde je možnost určit, zda může být daný objekt detekovatelný proximity senzorem, případně jakým typem
- Rendable - zde je možnost určit, zda může být daný objekt detekovatelný vision senzorem
- Viewable - toto nastavení určuje, zda objekt může být zobrazen v dalších pohledech

2.4 Nastavení objektů [2]

Každý z objektů má poměrně široké možnosti nastavení. Do přístupu k těmto nastavením je nutný dvojklik na objekt ve stromové struktuře scény, případně označením objektu a užitím nabídky: Menu bar/Tools/Scene Object Properties.

Všechny objekty mají obecné nastavení vlastností, stejné pro všechny objekty, a také speciální nastavení, které je vlastní jen pro daný objekt. Např. vazby mají jiné speciální nastavení než Tvary nebo Grafy. Mezi obecné nastavení patří zejména to, v jaké zobrazovací vrstvě bude objekt viditelný, dále zda bude detekovatelný, měřitelný, případně měřítko jeho velikosti, atd.

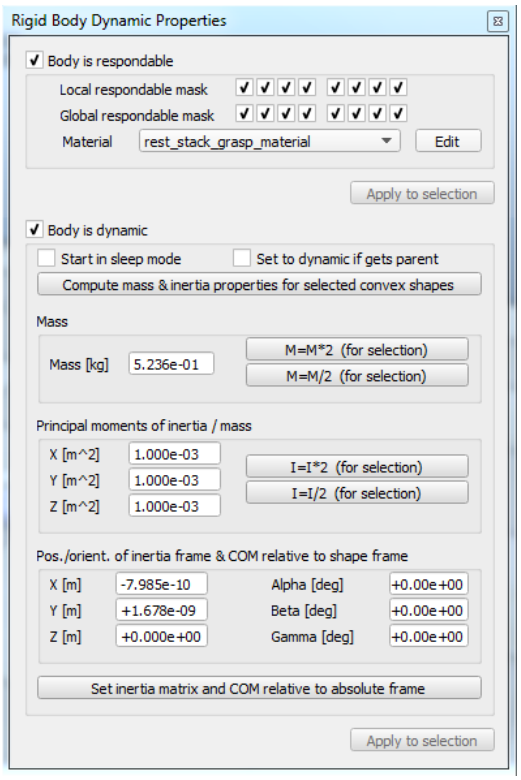
Popis speciálního nastavení pro vybrané objekty:

- tvary - U tvarů je možné měnit rozměry, barvu, přidávat textury, měřítko, a zejména měnit dynamické nastavení tvaru. Je možné určit, zda objekt má být dynamický, a k tomu příslušející hmotnost a momenty setrvačnosti k jednotlivým osám. Dále zda má být reaktivní, neboli zda má docházet ke kolizím při střetu s jinými objekty, případně z jakého materiálu je daný objekt tvořen. Dynamické nastavení tvaru lze vidět na obrázku 2.4.

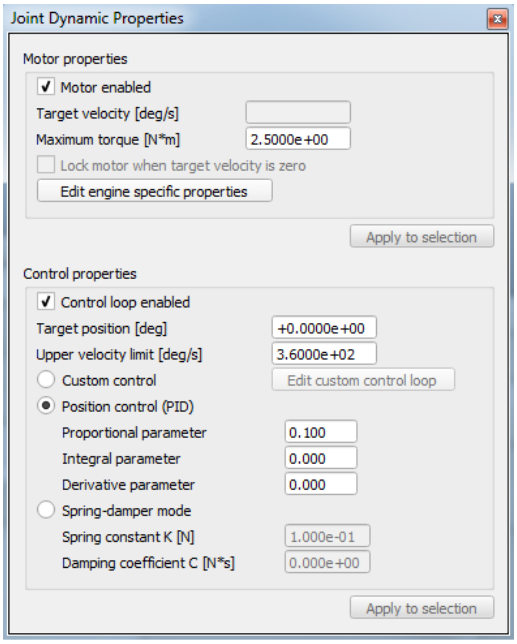
- vazby - U vazeb lze nastavovat zejména jejich omezení, případně zda jde o cyklické opakování pohybu v rámci vazby. Další podstatné nastavení je tzv. mód vazby, který určuje její chování, např. dynamický mód, pasivní mód, mód inverzní kinematiky, atd. V rámci této diplomové práce jsou všechny užité vazby v dynamickém módu. V případně dynamického módu lze dále nastavovat, zda má mít vazba nějaký pohon, případně zda tento pohon má mít nějakou regulaci. Dynamické nastavení vazby lze vidět na obrázku 2.5.

- dummy - Dummy se nejčastěji používá jako propojení ve stromové struktuře. V případě aktivního propojení dvou dummy je třeba mít nastavený dynamický režim a také je třeba dávat pozor na rodičovské prvky, ke kterým je dummy navázáno. Nejčastěji se váže tvar s vazbou, kdy se jedná o pohyblivý spoj, případně je také možné spojit tvar s tvarem. Výsledkem je pak vazba pevná.

Některé druhy objektů potřebují ke své správné funkci tzv. Floating view. Jedná se o druh okna vloženého do simulace, na kterém je zobrazena činnost daného objektu. Zejména jde o zobrazení dat pomocí grafů, zobrazení obrazu z vizuálních senzorů nebo kamery. V dané scéně může být umístěno libovolné množství oken.



Obr. 2.4: Dynamické nastavení tvaru

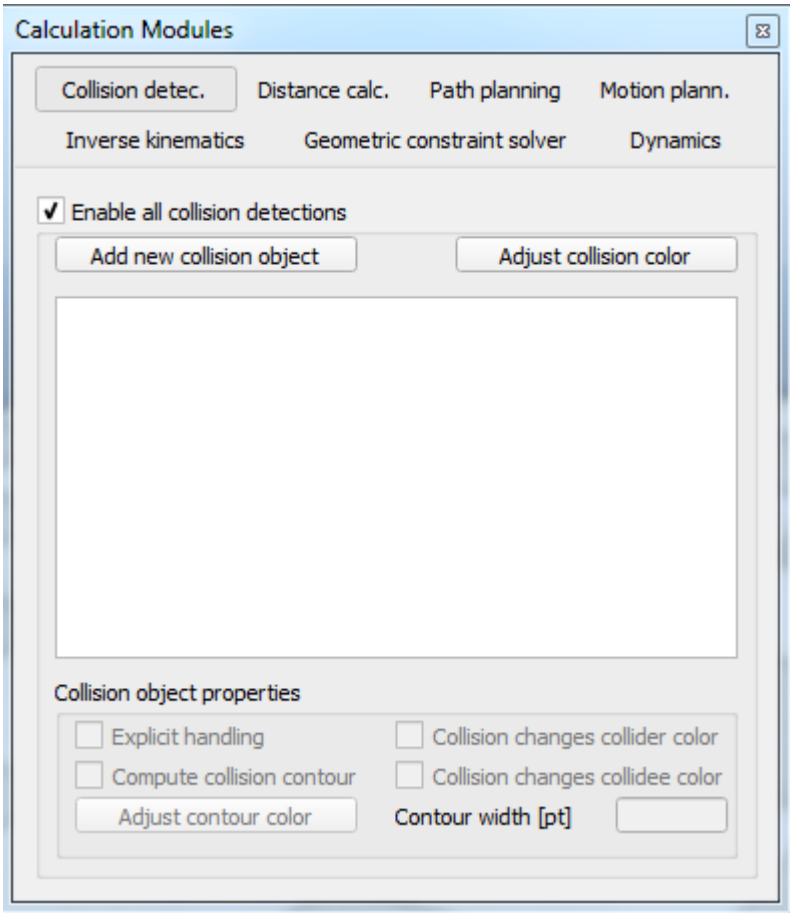


Obr. 2.5: Dynamické nastavení vazby

2.5 Prostředí scény a výpočetní moduly [2]

Prostředí ve V-REP je možné nastavovat v nabídce: Menu Bar/Tools/Environment. V této nabídce je možnost měnit například barvu pozadí, osvětlení scény, povolení kolizí, povolení textur, atd.

Další velmi užitečná věc jsou výpočetní moduly, díky kterým je možné získávat doplňující informace o scéne, případně o samotné simulaci. Jedná se zejména o detekci kolizí mezi objekty, výpočet vzdáleností, plánování trajektorie, nastavení pro inverzní kinematiku, výpočty geometrických omezení a dynamiky. Nabídka výpočetních modulů je zobrazena na obrázku 2.6. (Je ale podstatné dodat, že od nejnovější verze V-REP již modul pro Path planning a Motion planning není doporučované používat, jelikož byla vytvořena možnost využití pluginu OMPL. Ten je pro podobné výpočty mnohem silnějším a flexibilnějším nástrojem.)



Obr. 2.6: Nabídka výpočetních modulů

2.6 Programovací možnosti [2]

Základem každé scény vytvořené ve V-REP je takzvaný Main script, který obsahuje veškeré podstatné nastavení scény, a je nutný pro správný běh simulace. Výchozí nastavení tohoto skriptu je doporučené měnit pouze pokročilým uživatelům.

Další možností je vytvořením takzvaného Child script, který slouží již jako programovatelné prostředí pro danou simulaci. Veškeré programovaní je realizováno v programovacím jazyce LUA. Tento druh skriptu lze vytvořit ke každému objektu, z čehož vzniká možnost vytvoření několika různých skriptů v rámci jedné scény. Pro efektivní práci s Child script jsou ve V-REP implementovány API functions. Jedná se o již vytvořené funkce pro nastavení nebo změnu téměř všech parametrů v dané scéně.

Další možností je využití jiných programovacích jazyků. Jsou to C/C++, Python, Java, Matlab, Octave, Lua, Urbi. Pro tyto jazyky jsou vytvořeny Remote API function, které slouží v těchto jazycích podobně jako již zmíněné API function ve V-REP. V tomto případně je také nutné nastavit parametry komunikace s aplikací daného jazyka. Existují i další možností programování ve V-REP, zde byly zmíněny pouze ty nejjednodušší a nejpoužívanější.

2.7 Simulace [2]

Správné nastavení parametrů simulace je zásadní pro relevantnost zobrazovaných výsledků. Mezi nejdůležitější nastavení patří zvolení vhodného řešiče dynamiky.

Momentálně je možnost si vybrat ze 4 řešičů:

- Bullet Physics Library - open source dynamický řešič, zahrnující 3D kolize, Rigid body dynamics, Soft body dynamics. Často používaný jako herní fyzikální engine.
- Open Dynamics Engine (ODE) - open source dynamický řešic, zahrnující 3D kolize, Rigid body Dynamics. Často používaný jako herní fyzikální engine.
- Vortex Dynamics Engine - Closed source. Placený řešič.
- Newton Dynamics Engine - V součastnosti v beta verzi. Široko-platformový fyzikální engine.

Nejčastěji používaně jsou díky licenci open source řešiče Bullet a ODE.

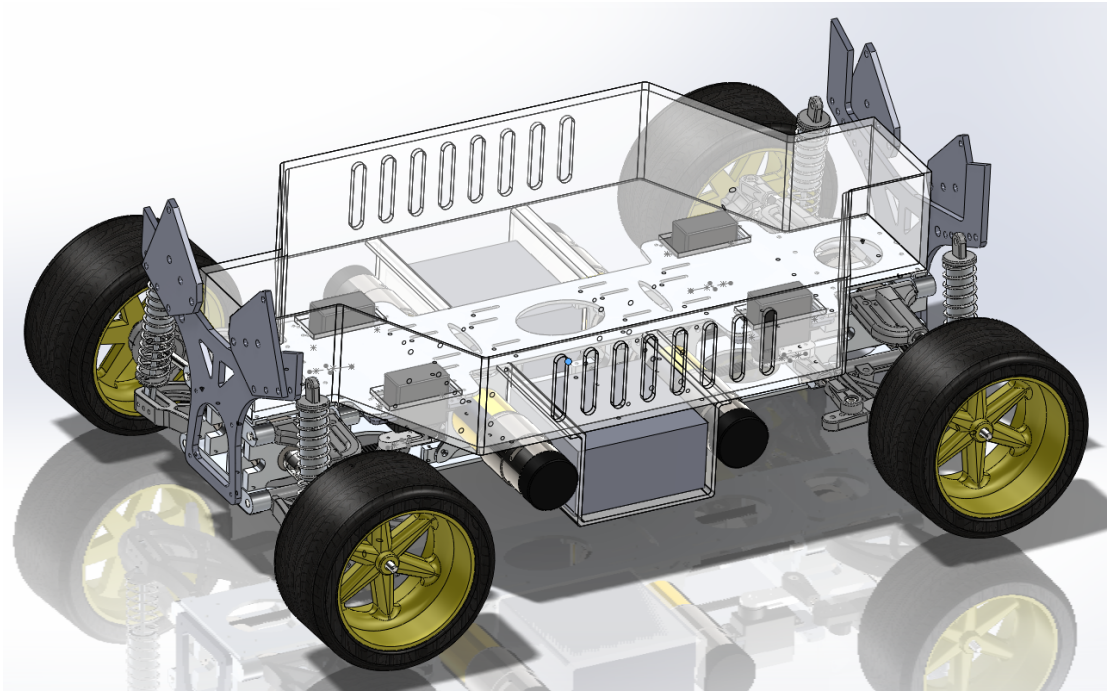
Další velmi důležitou volbou je délka kroku simulace. Lze volit od maximální hodnoty dt = 200 ms po minimální hodnotu dt = 10 ms. Toho je potřeba nastavit o ohledem na náročnost scény a požadovanou přesnost výpočtů. S tím souvisí i nastavení přesnosti simulace, od Velmi rychlé, až po Velmi přesnou. Je také možnost nastavit danou simulaci jako real-time simulaci.

3 MODELOVÁNÍ AUTA MODELU CAR4

3.1 Úprava CAD modelu v software Solidworks a import do V-REP

V první fázi bylo přizpůsobeno několik dílů z modelu v Solidworks do současného stavu, jako například plechy pro uchycení senzorů, nárazník z plechu nebo vzhled servomotorů, které byly od původní verze vyměněny.

Poté bylo nutné již vytvořený model s Solidworks upravit takový způsobem, aby bylo možné díly co nejjednodušeji importovat do V-REP. Z toho důvodu bylo vytvořeno několik podsestav, ze kterých se jednotlivé díly snáze exportovaly do formátu .stl, což je jeden z formátů, které lze importovat do V-REP.

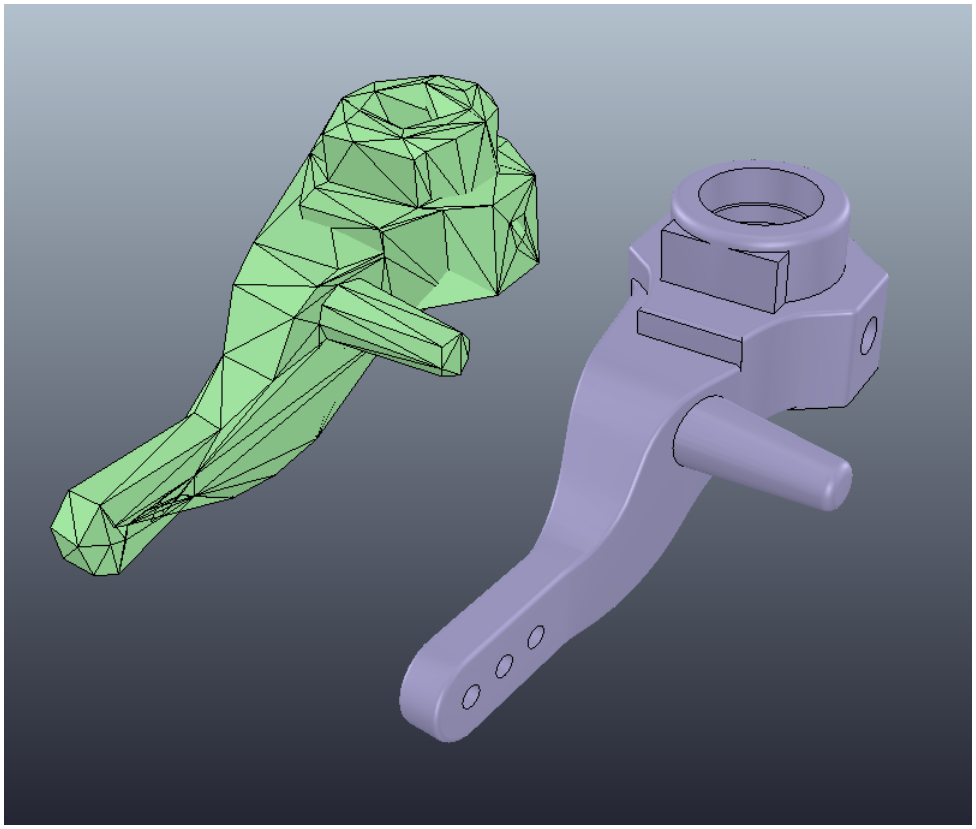


Obr. 3.1: Model v Solidworks

Pro Solidworks verze 2012 je dostupné zdarma rozšíření, umožňující převést sestavy do formátu URDF neboli „Unified Robot Description Format“. Jedná se o univerzální formát pro popis robotických systémů z knihovny ROS, která obsahuje Open Source nástroje a knihovny pro tvorbu robotických aplikací. V tomto formátu je již vytvořena stromová struktura modelu, včetně potřebných parametrů. Pro každou část modelu jsou vytvořeny dvě části, výpočtová, jenž je zjednodušená a slouží pro početní operace, a vizuální, která obsahuje všechny detaily a slouží pouze pro

samotné zobrazení součástí. Rozšíření je vhodné zejména pro jednodušší sestavy, neboť u složitějších hrozí vytvoření špatných vazeb mezi objekty, případně je nutné tyto vazby nastavovat ručně. Největším pozitivem je rozmístění dílčích objektů v prostoru, což je jeden z úkonů, které jsou ve V-REP dosti časově náročné.

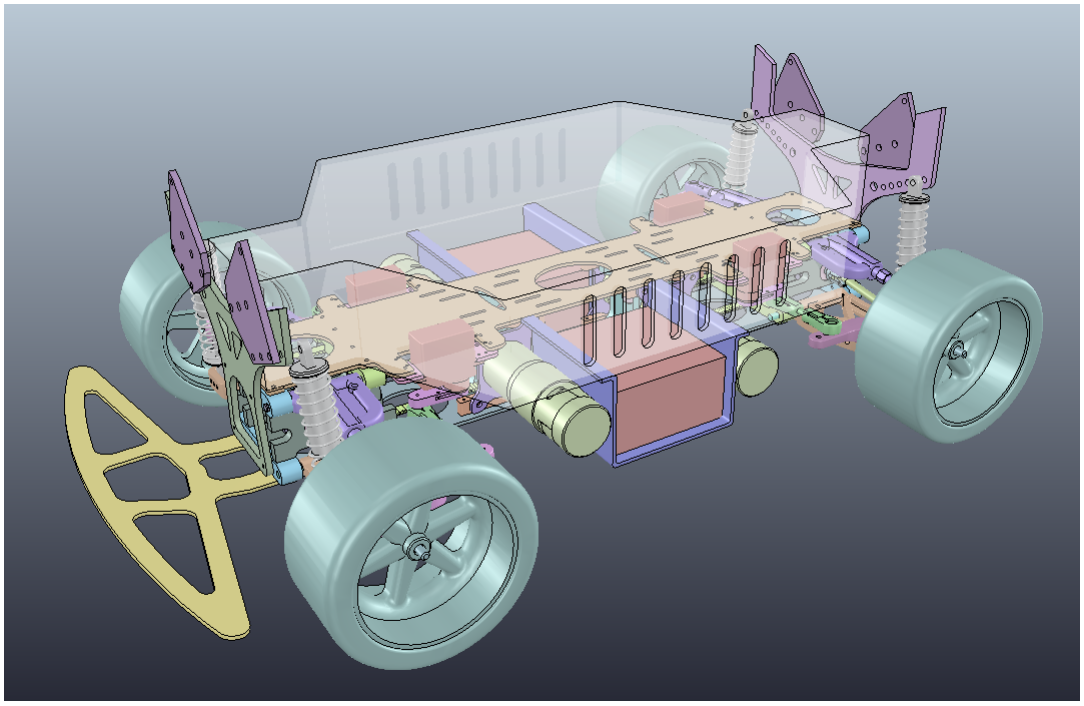
Po importu se díl, případně sestava, zobrazí ve středu absolutního SS a je potřeba jej umístit do žádané polohy pomocí souřadnic, jak bylo zmíněno v kapitole 2.2. Jelikož V-REP neumí pracovat se složitými tvary, je vhodné každý díl rozdělit na dvě části, výpočtovou a vizuální. Ve výpočtové části bude model zjednodušený tak, aby parametry odpovídal co nejvíce původnímu dílu, ale zároveň, aby co nejméně zatěžoval výpočet. Je tedy nutné experimentálně vyvážit poměr mezi zjednodušením a přesností. Ideálním případem je vytvoření dílu pomocí jednoduchých tvarů, jež nabízí základní nabídka programu. V případě složitějších tvarů je možné využít nástroje Vizual Decompozition, který díl zjednoduší do žádaného konvexního tvaru tak, že vytvoří trojúhelníkovou mesh. Nástroj nabízí i změnu parametrů pro dekompozici, z čehož je možné určit potřebnou míru zjednodušení. Ve vizuální části je naopak snahou, co nejvíce se přiblížit původnímu vzhledu součásti, jelikož slouží jen jako vnější skořepina výpočtové části, bez jakékoliv jiné interakce.



Obr. 3.2: Využití nástroje Vizual Decompozition - Těhlice kola

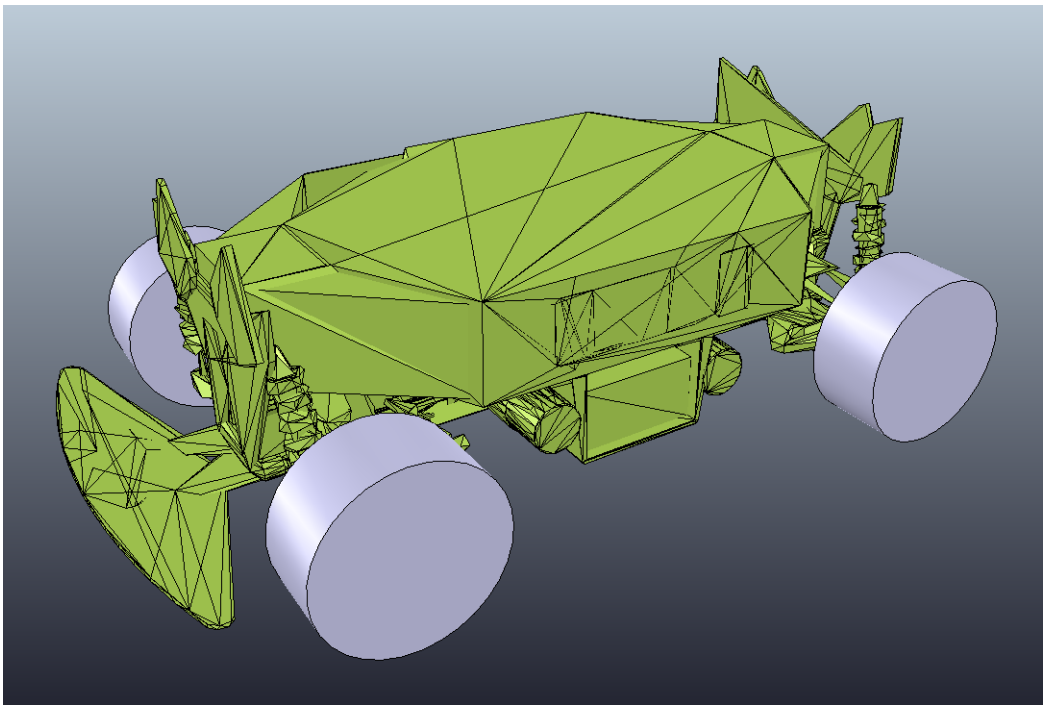
3.2 Tvorba jednoduchého modelu

Tento model je ukázka, jak je možno ve V-REP udělat co nejjednodušší model auta. Je tvořen pouze rámem vozu, a jednotlivými koly a vazbami nahrazujícími odpružení, zatáčení a pohon. Jelikož mezi dvěma objekty může být definována vždy jen jedna vazba, bylo nutné vytvořit i virtuální díly, které mají za úkol pouze přenést moment. Pevnou část auta tvoří rám, který byl modelován v rámci složitějšího modelu, což je důkladněji popsáno v kapitole 3.3. Na rám je připevněna sestava kola pomocí vazby Prismatic, která simuluje odpružení vozu. Na sestavu kola je následně uchycen virtuální díl samotného kola pomocí vazby Revolute, která simuluje zatáčení. Nakonec je již připevněno samotné kolo pomocí vazby revolute, která simuluje pohon vozu. Použité vazby jsou zobrazeny a popsány na obrázku 3.5. Na obrázku 3.3 lze vidět vizuální model auta, který byl vytvořen v rámci složitého modelu a je podrobně popsán v podkapitole 3.3.1. Na obrázku 3.4 je zobrazen výpočtový model auta.

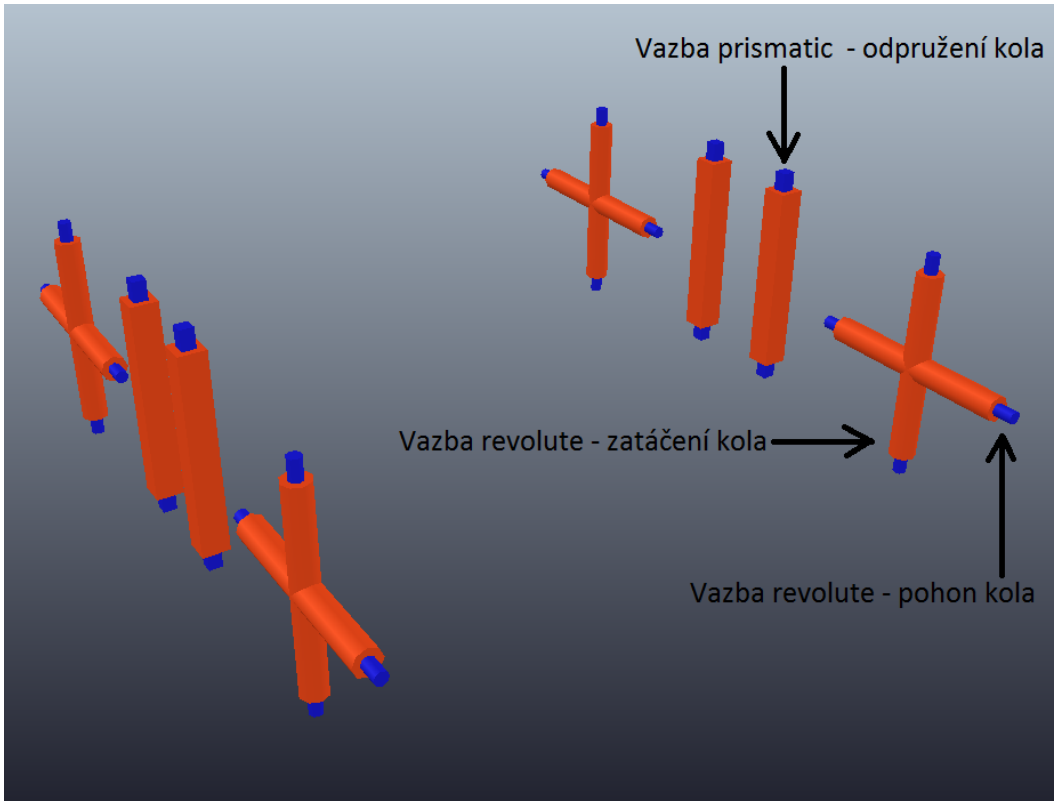


Obr. 3.3: Model vizuální zjednodušený

Vazby pro zatáčení a pohon jsou řízeny pomocí skriptů v integrovaném programovacím prostředí pomocí jazyka LUA, případně pomocí programu Matlab, a vše je podrobně rozebráno v kapitole 4.



Obr. 3.4: Model výpočtový zjednodušený



Obr. 3.5: Vazby zjednodušeného modelu

3.3 Tvorba složitého modelu

U složitého modelu probíhalo modelování komplexněji. Motivací bylo vytvořit model co nejvíce podobný reálnému modelu auta, zejména v oblasti parametrů jednotlivých dílů a s využitím reálných vazeb mezi pohyblivými díly. Vzhledem k tomu, že u reálného modelu jsou všechna kola řízena zvlášť, je také jejich uchycení k rámu a připojení k pohonu řešeno jinak, než je tomu běžné u klasických automobilů. To je znát zejména absencí jakéhokoliv propojení mezi levým a pravých kolem.

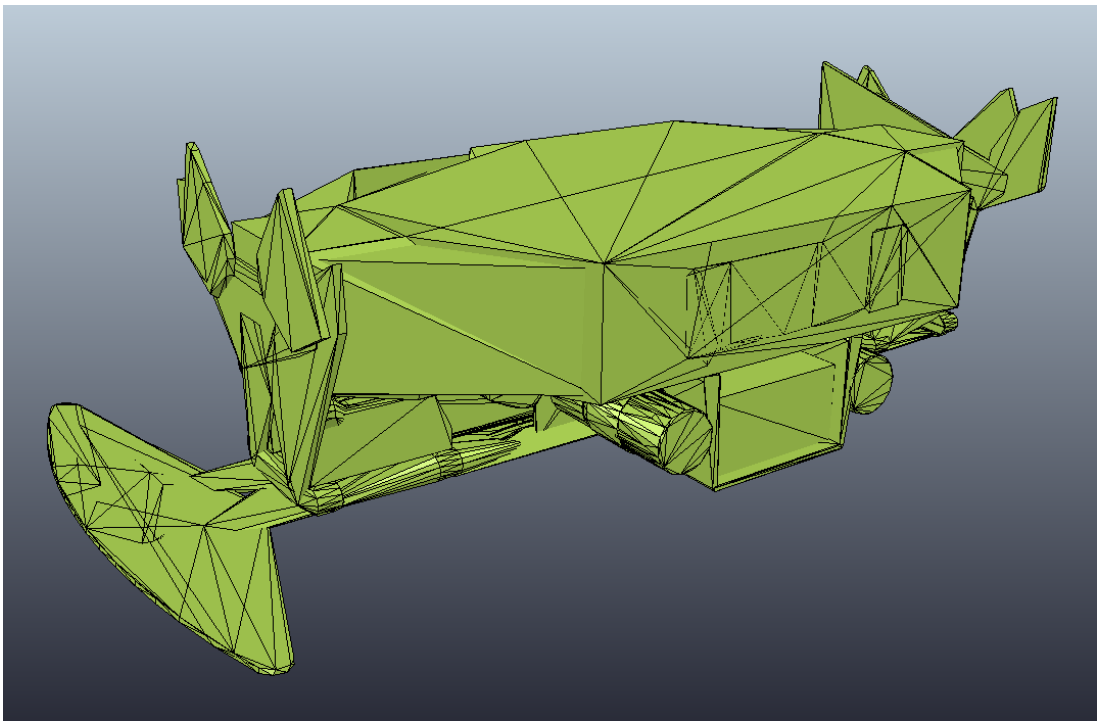
Modelování probíhalo ve třech fázích:

- modelování rámu
- modelování levého a pravého kola
- připojení kola k rámu

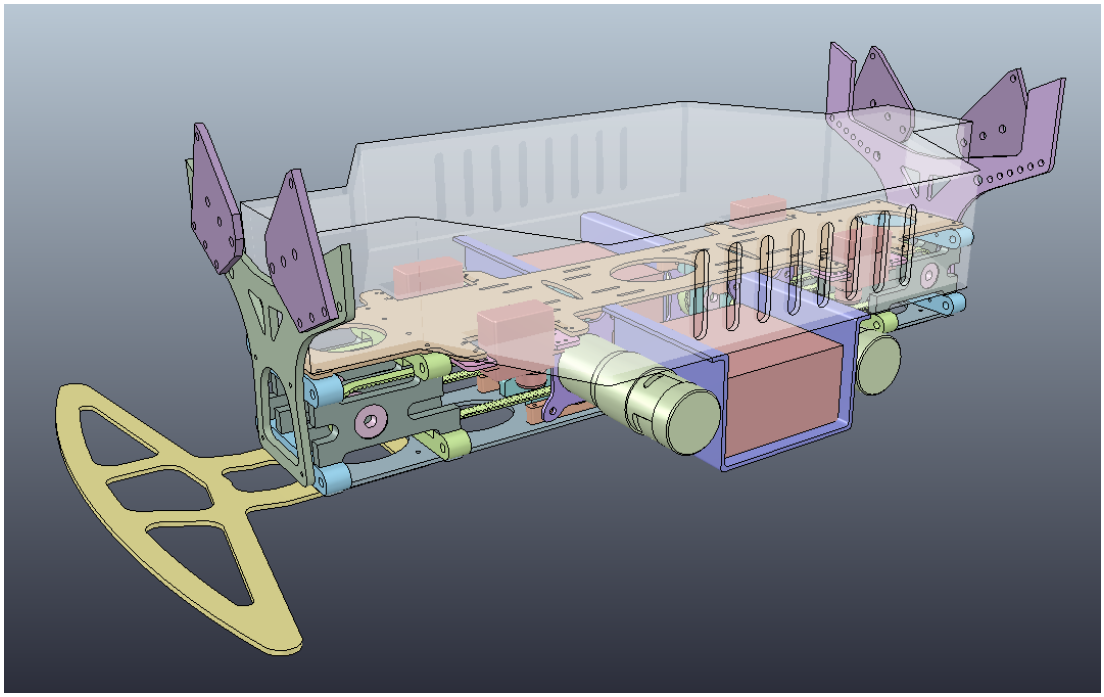
3.3.1 Modelování rámu

Aby se výpočtový model rámu co nejvíce blížil realitě, muselo by probíhat modelovaní skládáním všech dílů do výsledné sestavy. V takovém případě by bylo nutné u výpočtových modelů jednotlivých dílů nastavovat zvlášť hmotnosti a momenty setrvačnosti, což by vedlo k vysoké výpočtové náročnosti simulace bez většího reálného přínosu. Dále by bylo nutné zjistit přesné parametry všech součástí. Vzhledem ke statické povaze rámu se tento přístup ukázal jako neefektivní. Z tohoto důvodu byl výpočtový model rámu vytvořen pouze z jednoho dílu, což značně zjednoduší výpočty dynamiky a zároveň zajistí efektivnější manipulaci. Rámu složeného pouze z jednoho dílu se podařilo dosáhnout exportováním jeho celku ve formátu .stl z programu Solidworks. Následně byla použita funkce Vizual Decompozition, která byla popsána v kapitole 3.1. Výpočtový model rámu je zobrazen na obrázku 3.6

Na výpočtový model rámu byl v další fázi navrstven vizuální model rámu, který tvoří jednotlivé díly samostatně vkládané pomocí souřadnic na správné místo v sestavě. Tento postup se ukázal jako poměrně zdlouhavý, neboť bylo nutné odměřovat vzdálenosti jednotlivých dílů přímo v Solidworks. Bohužel jinou možnost manipulace s objekty není možné využít, což se dá označit jako nevýhoda programu V-REP. Manipulace s objekty a případné nedostatky v tomto směru byly popsány k kapitole 2.2.



Obr. 3.6: Výpočtový model rámu

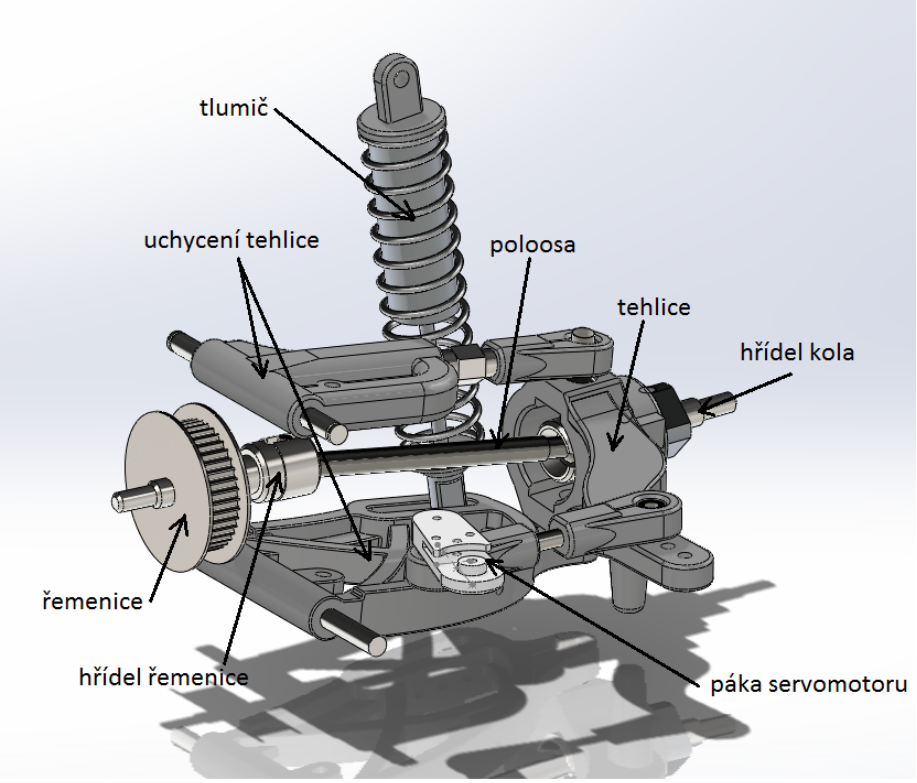


Obr. 3.7: Vizuální model rámu

3.3.2 Modelování kola

V modelu kola je maximální snaha o využití vazeb tak, jak je tomu u reálného modelu. Vzhledem k tomu, že výpočtové modely jsou tvarově zjednodušené, je realizace veškerého kontaktu mezi veškerými částmi kola tvořena pouze pomocí vazeb. Všechny díly jsou nastaveny jako dynamické, neboli mají vlastní hmotnost a moment setrvačnosti, ale zároveň jsou nastaveny jako nereaktivní, což znamená, že nejsou schopny interakce s jiným objektem pomocí vzájemné kolize. Jedinou výjimkou je samotná pneumatika, která interaguje s vozovkou nebo terénem, a má tedy nastaveny parametry tření.

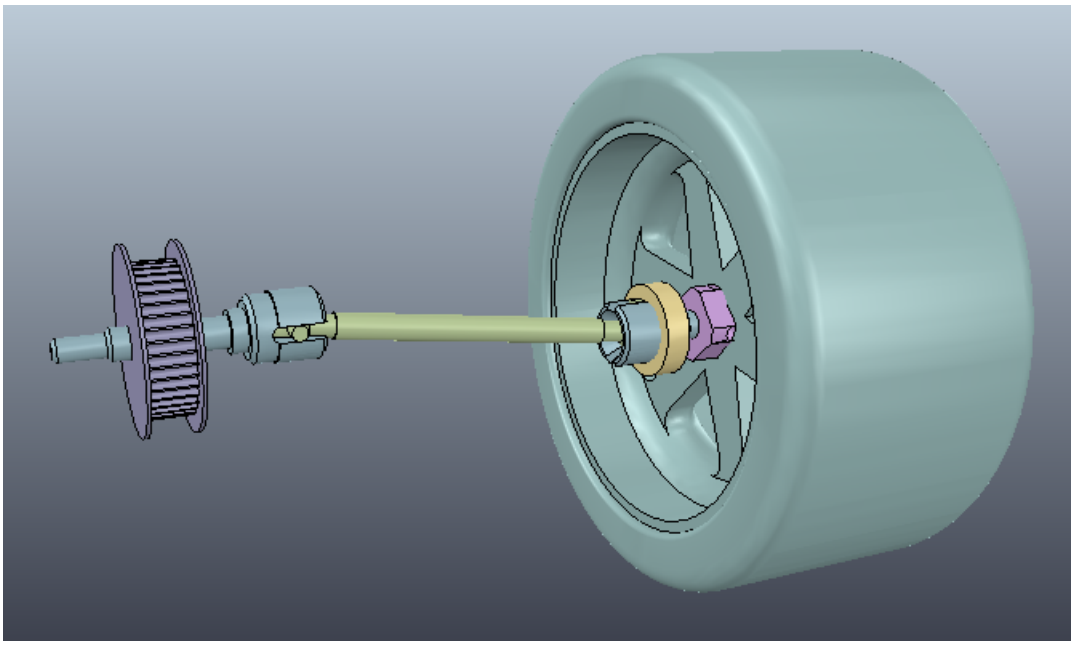
Na obrázku 3.8 je zobrazen model uchycení kola spolu se všemi díly, které obsahuje.



Obr. 3.8: Konstrukce kola

Veškeré vazby mezi jednotlivými díly jsou modelovány jako dynamické. Vzhledem k zjednodušení dílů ve výpočtovém modelu není možné povolit kolize mezi součástmi, což by vedlo k nefunkčnosti modelu. Toto lze vyřešit využitím vazeb s omezeným rozsahem. U vazby pohonu řemenice musel být ve rámci vazby aktivován cyklický mód, jinak by docházelo k nesprávné reakci vazby.

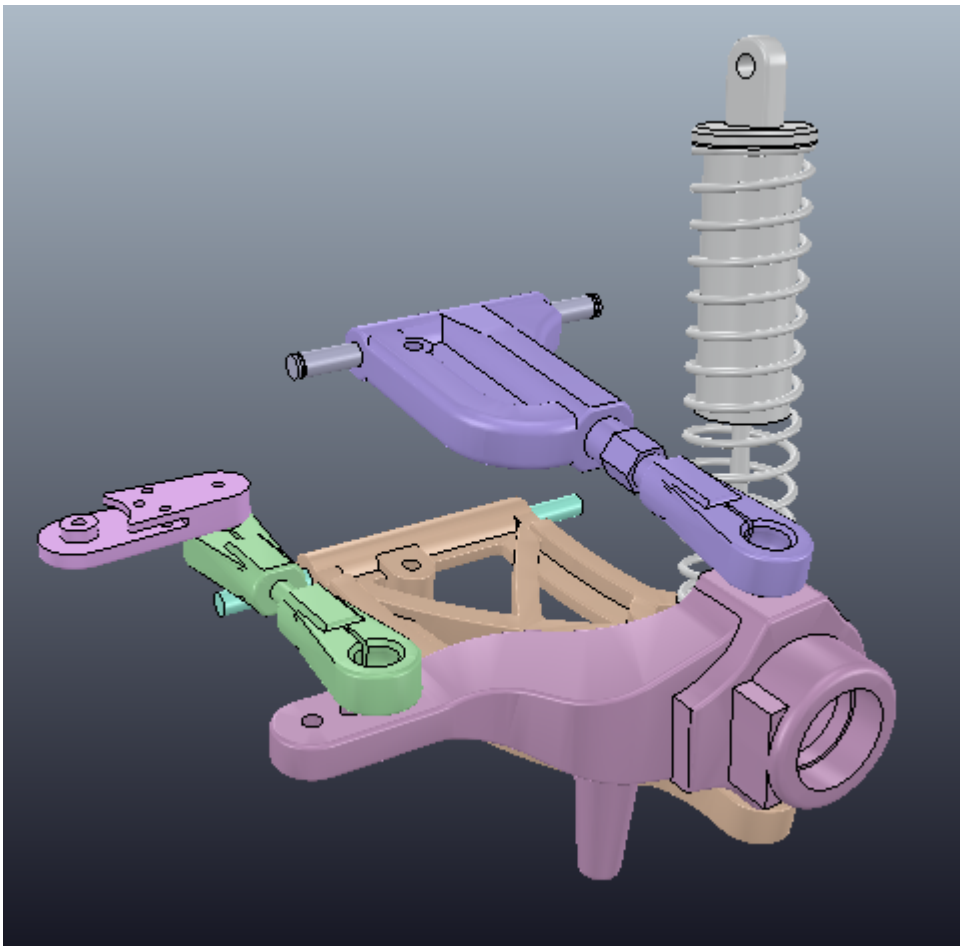
Model je možné rozdělit na dvě části. První část přenáší moment od řemenice, přes hřídel řemenice, poloosu, hřídel kola až na kolo samotné, viz. obrázek 3.9. Byly



Obr. 3.9: Namodelované součásti zajišťující přenos momentu od řemenice po kolo

zde využity pouze vazby přenášející kontakt, což se nejlépe realizuje posuvnou vazbou prismatic s nulovým posunem orientovanou kolmo k ose rotace. Druhá, o něco složitější část, se skládá zejména z těhlice, ve které je pomocí ložiska uchycena hřídel kola 3.10. Ta je připevněna pomocí kulových čepů, které jsou reprezentovány sférickými vazbami, ke dvěma nad sebou položeným ramenům, uchycením k rámu otočnými čepy (rotační vazba). Mezi rámem a těhlicí je dále umístěn tlumič, tvořený posuvnou vazbou prismatic s definovanou zpětnou vazbou ve formě tlumení. Bylo tedy možno k této vazbě definovat konstanty tlumení b a tuhost k, což nám zaručí výsledné odpružení celého vozu, podobně jako u reálného experimentálního vozidla. Nakonec je těhlice uchycena pomocí ramene s dvěma kulovými čepy k servomotoru, který pomocí rotační vazby mění celkové natočení kola.

Vzhledem k tomu, že V-REP neumí provázat dva díly mezi sebou s využitím dvou vazeb, bylo nutné u modelování kontaktu mezi hřídelí řemenice / poloosou a poloosou / hřídelí kola využít tzn. virtuálního dílu, neboť v místě styku dochází ke dvěma rotačním pohybům. Tento díl funguje pouze jako pomocný, ale pro správnou funkci musí mít také přiřazenou jistou hmotnost a momenty setrvačnosti. I díky tomu může docházet k mírnému zkreslení při výpočtech řešiče, neboť tento díl nemá žádnou reálnou předlohu.



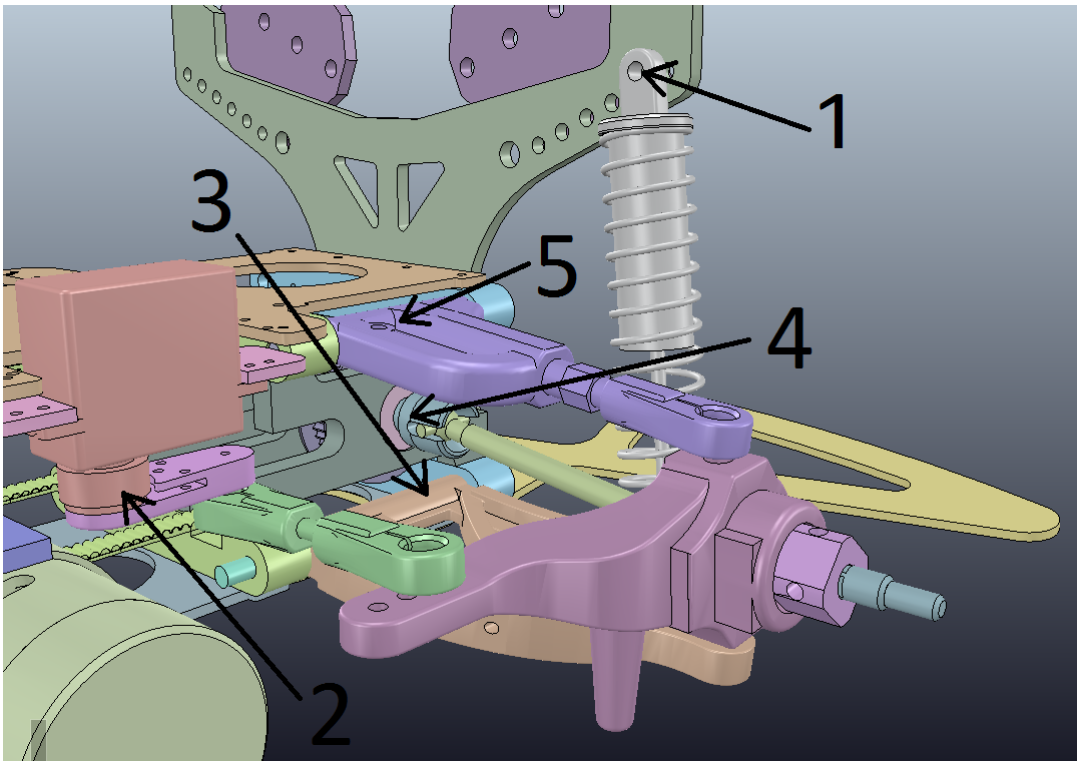
Obr. 3.10: Konstrukce pro zavěšení kola

3.3.3 Připojení kola k rámu

Připojení kola k rámu je realizováno na pěti místech 3.11. Na dvou místech jsou připojena ramena, jež drží těhlici, k rámu pomocí otočného čepu. Další tvoří připojení servomotoru. Následně je mezi rámem a kolem umístěno odpružení, a poslední je uchycení řemenice s umístěním ložiska. Bylo nutno přizpůsobit spoustu parametrů, neboť při zatížení kola rámem docházelo k mnoha problémům s vazbami. Totéž se stávalo při přivedení momentu na kola, případně při realizaci maximálního natočení kola.

Pro pevné spojení několika dílů je možné ve V-REP využít několika způsobů. Nejefektivnější způsob je vybrat objekty, které je nutné spojit a použít příkaz Group. Tímto se ze spojovaných dílu stane jeden díl. Spojení rámu a kol v jeden díl by však popřelo funkčnost výsledného modelu. Další způsob je využití objektu Force senzor, který dokáže nejenom měřit silovou interakci mezi tělesy, ale také vytvoří pevnou vazbu. Nevýhodou však je nutnost použití přímo mezi dvěma objekty. Vzhledem ke

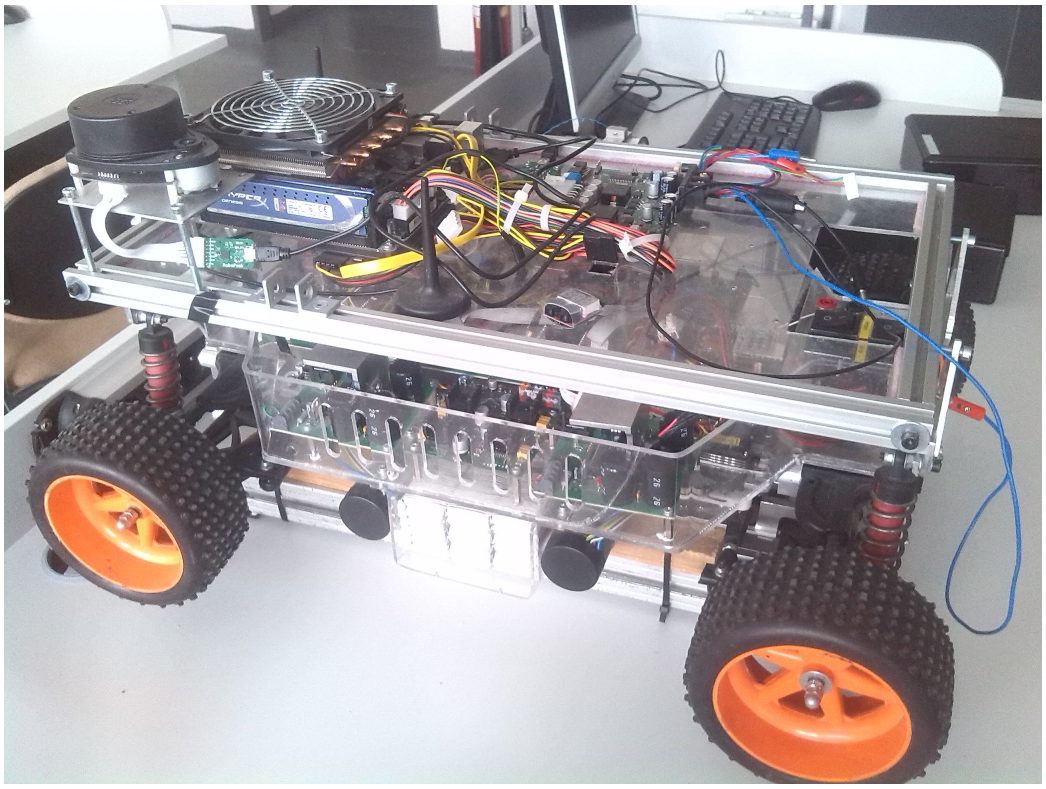
složitosti stromové struktury modelu není tento přístup reálně použitelný. Poslední způsob je využít spojení dvou objektů Dummy, které dokáží propojit dvě různá místa ve stromové struktuře modelu. Tímto způsobem bylo nejefektivněji realizováno propojení modelu uchycení kola s rámem vozu.



Obr. 3.11: Uchycení kola k rámu na pěti místech

3.3.4 Porovnání s reálných modelem

Jak již bylo zmíněno, byla snaha vytvořit model co do funkčnosti nejvíce podobný reálnému experimentálnímu vozidlu. Vzhledem k omezením plynoucím z hardwaru použitelném v dnešní době a výpočtové komplexnosti dynamiky v reálném čase bylo nutné přistoupit k jistým zjednodušením. Veškeré aktivní díly byly použity ve zjednodušené formě. Z tohoto důvodu hmotnosti a momenty setrvačnosti jednotlivých dílů jsou pouze přibližné. U modelování přenosu momentu z řemenice na kolo bylo nezbytné využití Virtuálních dílů. Tyto součásti umožnily funkční propojení podobné realitě, avšak za cenu jistého zkreslení při výpočtu. Jistou komplikací bylo také využití sférických vazeb v místech kulových čepů, neboť tento druh vazby ve V-REP není možno omezit. I přes výše zmíněné nedostatky se však podařilo zachovat princip pohybu na jednotlivých nápravách a také výsledný pohyb modelu vozidla vypadá reálně.



Obr. 3.12: Reálný model experimentálního vozidla CAR4

Obr. 3.13: Připojení kola k rámu na reálném vozidle CAR4

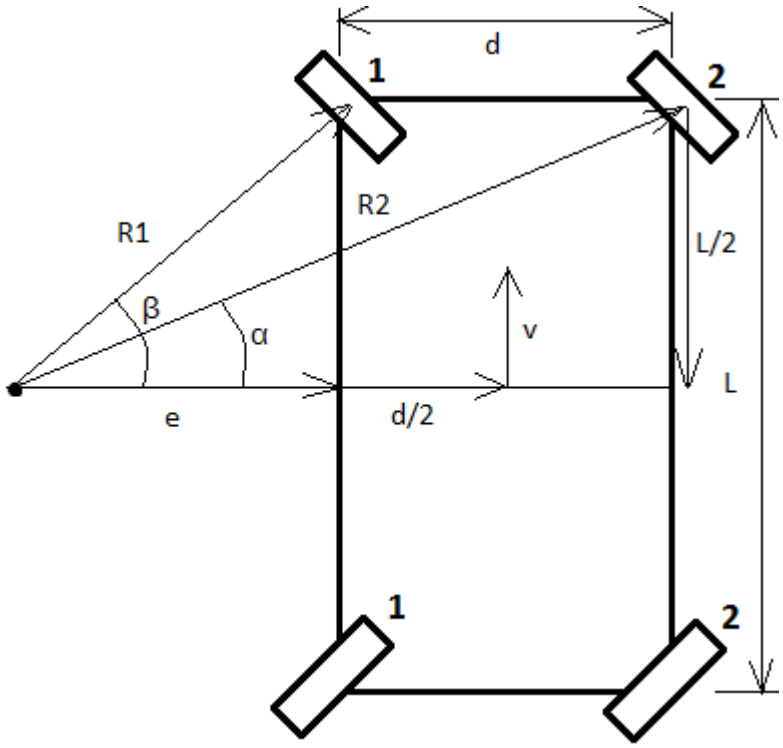
4 ŘÍZENÍ MODELU

4.1 Využití Ackermanovy kinematiky [1]

Princip Ackermanovy kinematiky pro vozidlo se čtyřmi samostatně řízenými koly byl podrobně popsán pro experimentální vozidlo CAR4 v diplomové práci [1], (kapitola 3.4.2, strana 42). Tento popis je však příliš komplexní pro využití v simulaci a proto je potřeba jej přizpůsobit pro užití v simulaci.

První předpoklad pro zjednodušení je ten, že natočení kol na stranách vozu bude vždy stejné. Dále se bude vycházet z předpokladu, že vždy známe úhel natočení na vnitřních nápravách (na straně vozu, která je blíže středu otáčení), s označením β. Dále známe délku vozu L, šířku vozu d a dopřednou rychlost v.

Z obrázku 4.1 lze odvodit zbytek proměných.



Obr. 4.1: Kinematika vozidla v simulaci

Výpočet vnějšího úhlu zatáčení:

$$\tan\alpha = \frac{\frac{L}{2}}{d+e}$$

$$\tan\beta = \frac{\frac{L}{2}}{e} => e = \frac{\frac{L}{2}}{\tan\beta}$$

$$\alpha = \arctan\frac{\frac{L}{2}}{d+\frac{\frac{L}{2}}{\tan\beta}}$$

Výpočet rychlostí kol:

$$R_1 = \frac{\frac{L}{2}}{\sin\beta}$$

$$R_2 = \frac{\frac{L}{2}}{\sin\alpha}$$

$$v_1 = v\frac{R_1}{\frac{d}{2}+e}$$

$$v_1 = v\frac{R_2}{\frac{d}{2}+e}$$

Se znalostí proměných α, β, v_1 a v_2 je možné vytvořit řízení celého modelu.

4.2 Řízení modelu užitím V-REP skriptu

K řízení modelů ve V-REP lze přistoupit několika různými způsoby, jak již bylo nastíněno v kapitole 2.6. Jako jednoduchá a velmi efektivní se jeví možnost využití Non-threaded Child script, který se chová jako funkce.

4.2.1 Non-Threaded Child script

Pokud je Non-Threaded Child script zavolán, vykoná nadefinované akce a následně vrátí hodnoty řízení do scény. Dále je charakteristický tím, že se částečně vykonává v každém jednotlivém kroku simulace.

Každý Non-threaded child script se skládá v základní podobě ze čtyř částí:

- Inicializace hodnot v simulaci - vykonává se pouze jednou, a to na počátku simulace
- Vykonání příkazů simulace - Vykonává se v každém kroku simulace, ve vykonávací části programu definované v Main Scriptu
- Snímací část simulace - Vykonává se v každém kroku simulace, ve snímací části programu definované v Main Scriptu
- Obnovovací část kódu - Vykonává se pouze jednou, a to těsně před ukončením simulace

Jakékoliv kusy kódu je však nutno psát pouze do výše zmíněných částí, které jsou automaticky vytvořeny spolu se scriptem, v opačném případě dojde k chybě. Toto je způsobeno tím, že jednotlivé části jsou definovány a spouštěny pomocí Main Scriptu.

V každé simulaci můžu být užito více Child scriptů. Child script je vždy spjat s některým objektem v dané scéně, kde je zobrazen jako ikona listu papíru. V rámci V-REP je možno také využít tzv. Threaded Child Script, ale vzhledem k nevýhodám nejsou využity v modelu experimentálního vozidla. Platí pravidlo, že dříve se vykonávají Child scripty přiřazené objektu, který je umístěn výše v hierarchii scény, neboli který má nejvyšší místo ve stromové struktuře scény.

4.2.2 Struktura řízení modelu

V případě simulace experimentálního vozidla byla využita pouze inicializační část a část vykonání příkazů, zbylé dvě části nebylo nutné použít.

Inicializační část Scriptu slouží zejména k definování základních konstant a parametrů modelu, případně k výpočtům, které je nutné vykonat pouze jednou v rámci celé simulace. V případě modelu CAR4 jsou to zejména rozměry vozu, dále nejrůznější omezení rychlosti a zatáčení, případně dalším výpočtům. Dále také slouží k definování jednotlivých proměných modelu, se kterými se bude pracovat ve vykonávací části simulace. U vozidla se jedná zejména o rychlosti jednotlivých kol a také o úhel natočení na servomotorech. K těmto proměným je nutné přiřadit ty vazby v modelu, které danou funkci obstarávají. V našem případě jde o vazbu Motor, spojenou s řemenicí a o vazbu Zataceni, spjatou s pákou servomotoru. Pro každé kolo jsou tyto proměné definovány zvlášť a k tomuto účelu slouží API funkce *simGetObjectHandle*.

```
steer_handle  = simGetObjectHandle('zataceni')
steer_speed   = simGetObjectHandle('motor')
```

Ve vykonávací části scriptu je definováno samotné řízení modelu. To probíhá pomocí vstupů z klávesnice, jako je tomu obdobně například u počítačových her. Ve V-REP jsou již definovány funkce, které obstarávají vstup z klávesnice. Je tedy pouze důležité definovat detekování takového signálu, a následné určit příslušnou reakci. Jako akční klávesy byly vybrány tradičně šipky. Šipky doleva a doprava obstarávají zatáčení, šipky dopředu a dozadu obstarávají pohon. Dále pomocí mezerníku je definováno zastavení vozidla. Nesmí byt zapomenuto na omezení maximálního úhlu natočení a maximální rychlosti u kol, jež byly definovány v Inicializační části.

Ukázka kódu pro vstup z klávesnice u dopředného pohybu vozidla:

```
message,auxiliaryData = simGetSimulatorMessage()
    while message ~= -1 do
        if (message == sim_message_keypress) then
            if (auxiliaryData[1]==2007) then
                speed=speed+d_speed
                    if speed >= max_speed then
                        speed = max_speed
                    end
            end
     end
end
```

Popis algoritmu:

Nejprve jsou přijímány informace ze simulace pomocí příkazu *simGetSimulatorMessage*. Následuje *while* cyklus, který se provede po přijetí zprávy. Následně se pomocí funkce ověří, zda se jedná o vstup z klávesnice. Pokud ano, porovná se, zda se jedná o klávesu „šipka dopředu“, která má označení 2007. Pokud vše souhlasí, dochází se zvýšení rychlosti, případně k její limitací při překročení maximální hodnoty.

V další části jsou s využitím Ackermanovy kinematiky, která byla zmíněna v kapitole 4.1,vypočtena jednotlivá natočení kol a také rychlosti otáčení. Je také možno definovat maximální moment, který může být využit k pohonu kol, ale v tomto případě je tak již učiněno v nastavení samotných vazeb.

V poslední fázi jsou údaje o rychlosti a úhlu natočení poslány zpět do jednotlivých vazeb a v dalších krocích simulace se tyto výpočty ve vykonávací části scriptu

opakují. K tomu složí funkce *simSetJointTargetPosition* a *simSetJointTargetVelocity*.

```
simSetJointTargetPosition(steer_handle,steeringAngle)
simSetJointTargetVelocity(steer_speed, speed)
```

4.3 Řízení modelu užitím Matlabu

Oproti řízení pomocí vestavěných scriptů ve V-REP má řízení užitím software Matlab svá specifika. Nejprve je nutné správně nastavit komunikaci obou programů. Následně je podstatné nastavit podmínky a způsob výměny dat mezi programy.

4.3.1 Nastavení komunikace mezi V-REP a Matlab [3]

Pro správnou komunikaci mezi oběma programy jsou zapotřebí tři soubory:

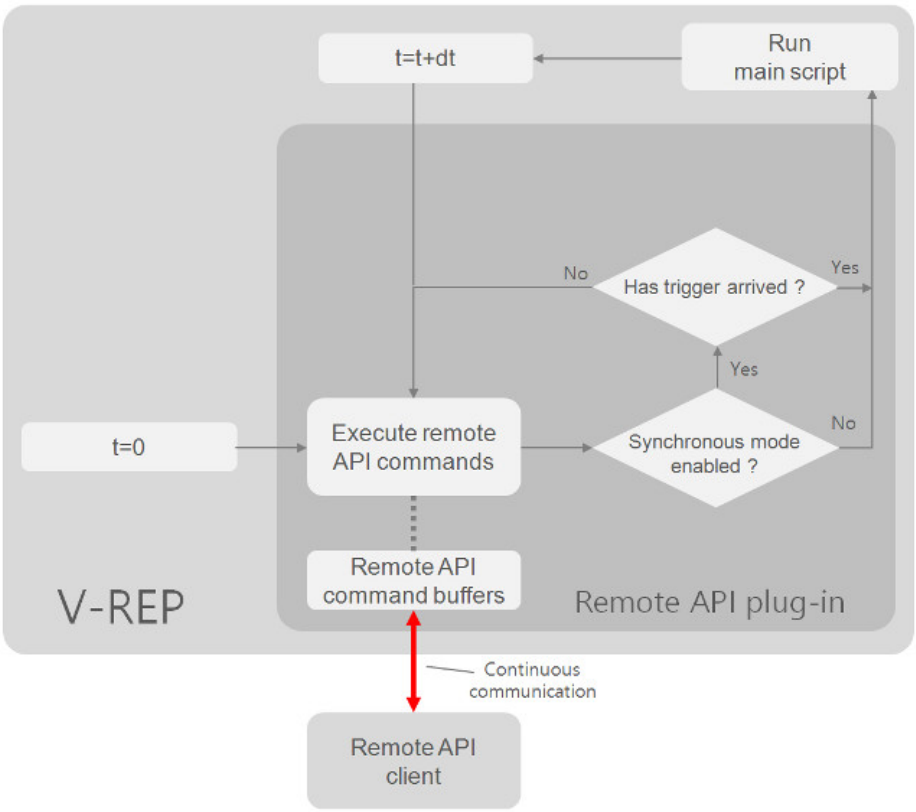
- remoteApiProto.m
- remApi.m
- remoteApi.dll (platí pouze pro operační systém Windows)

V souborech pro matlab remoteApiProto.m a remApi.m je definována většina funkcí potřebných pro řízení modelů. Soubor remoteApi.dll zajišťuje komunikaci pro daný operační systém, v našem případě pro Windows. V případě systému pro Mac OS X je nutné použít soubor remoteApi.dylib a v případě Linuxu se jedná o soubor remoteApi.so. Výše zmíněné soubory je nutné mít umístěné ve stejné složce, jako je uložena daná scéna ve V-REP a zdrojový M-file pro Matlab. Výše zmíněné soubory lze nalézt v instalační složce programu pod ...programming/remoteApiBindings/matlab.

Pro zajištění komunikace je taktéž nutné v instalační složce programu v souboru remoteApiConnections.txt povolit komunikaci V-REP s externím programem, a taktéž nastavit komunikační port. Při samotném psaní kódu je nezbytné jako první povolit užití remoteAPI funkcí a zároveň spustit komunikaci mezi programy užitím funkce *simxStart* . Zde se definuje IP adresa serveru a taktéž komunikační port a další nastavení pro správný přenos dat. Dále dojde ke spuštění samotné simulace funkcí *simxStartSimulation*.

```
vrep=remApi('remoteApi');
id = vrep.simxStart('127.0.0.1', 19997, true, true, 2000, 5);
ab = vrep.simxStartSimulation(id, vrep.simx_opmode_oneshot_wait)
```

Samotná výměna dat probíhá pomocí komunikačních socketů. Data jsou odeslána z Klient (Matlab) na Server (V-REP). Následně jsou vykonány dané příkazy a odeslána žádaná data zpět na Klient, viz obrázek 4.2. V ideálním případě Klient čeká na zpětnou vazbu od Server o vykonání daných příkazů. Vzhledem k tomu, že tento přístup by mohl neúměrně zpomalovat simulaci, je možné u každé funkce určit způsob, jakým má být vykonána.



Obr. 4.2: Komunikace mezi V-REP a externí aplikací [2]

Funkce mohou být spuštěny v několika pracovních módech:

- Blocking function calls - Při zavolání dané funkce se čeká na odpověď, zda byla funkce vykonána, až poté pokračuje další vykonávání programu.
- Non-blocking function calls - Při zavolání dané funkce se nečeká na odpověď, zda byla funkce vykonána a jsou posílány další příkazy..
- Data streaming - Užívá se zejména u funkcí pro sběr dat ze simulace. Je-li potřeba například znát pozici některé vazby, při jakékoliv změně této pozice jsou automaticky tyto data poslána do Matlabu.
- Synchronous operations - Tento režim se používá pro synchronizaci simulace ve V-REP s matlabem. Tímto lze zaručit, že veškeré příkazy jsou provedeny právě jednou v daném simulačním kroku. Vice je vysvětleno v kapitole 5.4.1.

Každá funkce taktéž vrací příznak, zda byla či nebyla provedena, případně z jakého důvodu provedena nebyla.

4.3.2 Struktura řízení modelu

Struktura řízení modelu je částečně podobná jako v kapitole 4.2.2. Nejprve je nutné definovat konstanty a parametry modelu. Toho se dosáhne funkcí *simxGetObjectHandle*. Dále se přiřadí prvky ze simulace (např. vazby, senzory), se kterými budeme pracovat, do proměných. Je zde však nutné správně definovat, jakým způsobem bude funkce zavolána. U inicializace je vhodné počkat na příznak, že vše bylo provedeno správně. K tomu je určen příkaz *simxOpmodeOneshotWait*.

```
[ab, steer_handle] = vrep.simxGetObjectHandle(id, 'zataceni' ,
vrep.simx_opmode_oneshot_wait)
[ab, steer_speed] = vrep.simxGetObjectHandle(id, 'motor' ,
vrep.simx_opmode_oneshot_wait)
```

V případě Matlabu již řízení neprobíhá pomocí vstupů z klávesnice, ale užitím joysticku. Matlab již obsahuje funkci *vrjoystick*, která obstarává komunikaci s joystickem. Je pouze nutné definovat žádoucí reakci při náklonu páčky v jednotlivých osách, případně při stisku některého z tlačítek. Toto je definováno klasickým způsobem, jak je běžné např. PC hrách. Při náklonu páčky v ose X dojde zatočení vozidla, pří náklonu v ose Y k rozjezdu, případně zpomalení.

Užitím Ackermanovy kinematiky jsou definována příslušná natočení kol v reakci na zatočení. V dalším kroku jsou užitím příslušných funkcí přeposlány informace o natočení kol a otáčkách do příslušných vazeb v dané simulaci. K tomu slouží příkazy *simxSetJointTargetPosition* a *simxSetJointTargetVelocity*. Zde je velmi podstatné zvolit správný příznak komunikace, kdy se nečeká na zpětnou vazbu, neboť by se tím několikanásobně zpomalila daná simulace. Proto se užije pouze příkaz *simxOpmodeOneshot*.

```
ab = vrep.simxSetJointTargetPosition(id,steer_handle,
steeringAngle,vrep.simx_opmode_oneshot)
ab = vrep.simxSetJointTargetVelocity(id,steer_speed
, speed,vrep.simx_opmode_oneshot)
```

5 MOŽNOSTI SOFTWARU V-REP

Software V-REP je velmi silným nástrojem pro modelování a simulaci v robotice. Obsahuje celou řadu zajímavých funkcí, jako například implementaci proximity senzorů, vizuálních senzorů, komunikaci s celou řadou externích programů, použití různých řešičů dynamiky, vizualizací dat a spoustou dalších. V této kapitole jsou shrnuty nejefektivnější nástroje vhodné pro využití v robotice.

5.1 Využití senzorů

V-REP obsahuje dva základní typy senzorů, které lze ve scéně využít. Proximity senzory, neboli senzory vzdálenosti, a vizuální senzory.

5.1.1 Proximity senzory

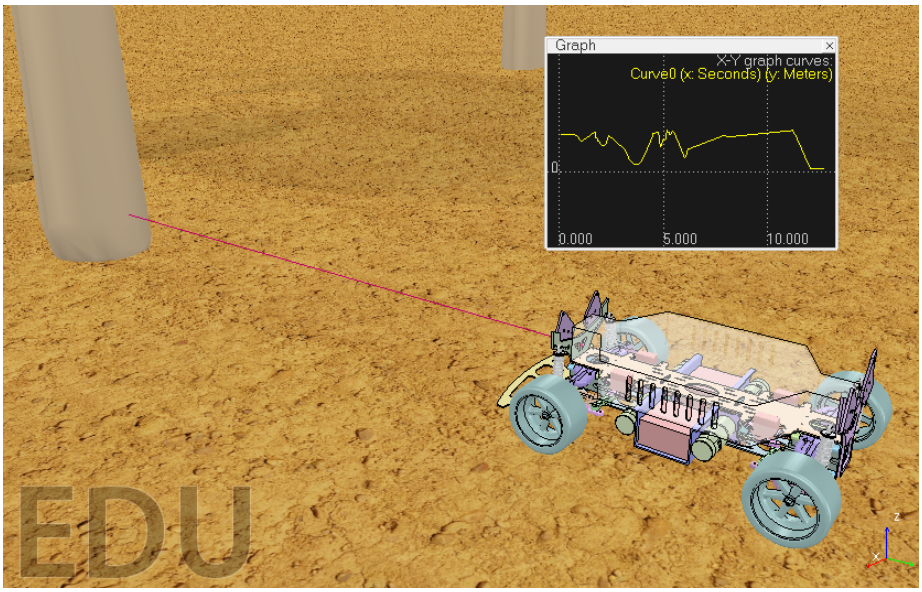
Existuje šest základních tvarů proximity senzorů:

- Ray type
- Randomized ray type
- Pyramid type
- Cylinder type
- Disc type
- Cone type

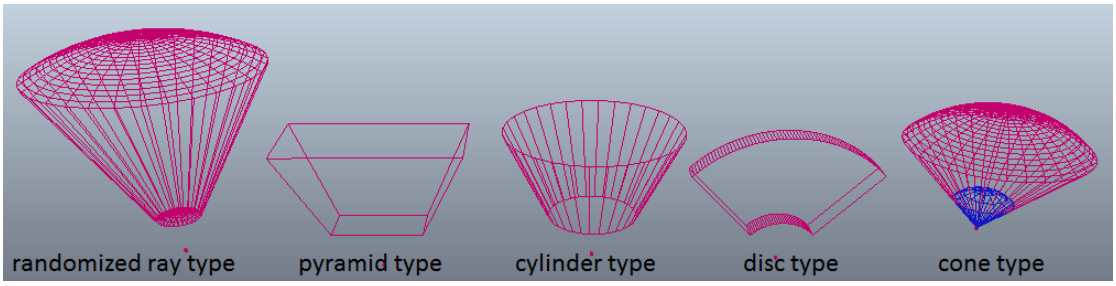
Ray type je nejjednodušší tvar senzoru a také nejpoužívanější. Je tvořen pouze jedním paprskem. Tento typ senzoru definuje pouze jeden parametr, a to dosah senzoru. Užití Ray type senzoru je zobrazeno na obrázku 5.1, kde lze také vidět zobrazení naměřených dat v čase užitím grafu.

Na obrázku 5.2 jsou zobrazeny další tvary senzorů. Nejdůležitějším parametrem pro jejich definování je dosah, dále pak úhel paprsků, počet paprsků, poloměr, atd..

U proximity senzorů je možnost volby, o jaký typ senzoru se jedná. U objektů pak následně bývá možnost volby, zda jsou detekovatelné pomocí senzorů, případně na který typ senzoru reagují.



Obr. 5.1: Užití ray type senzoru



Obr. 5.2: Tvarové typy senzorů

Na výběr je pět druhů senzorů:

- Ultrasonic
- Infared
- Laser
- Inductive
- Capative

Ve výchozím nastavení však typ senzoru nehraje žádnou roli, neboť všechny objekty jsou detekovatelné pomocí všech senzorů.

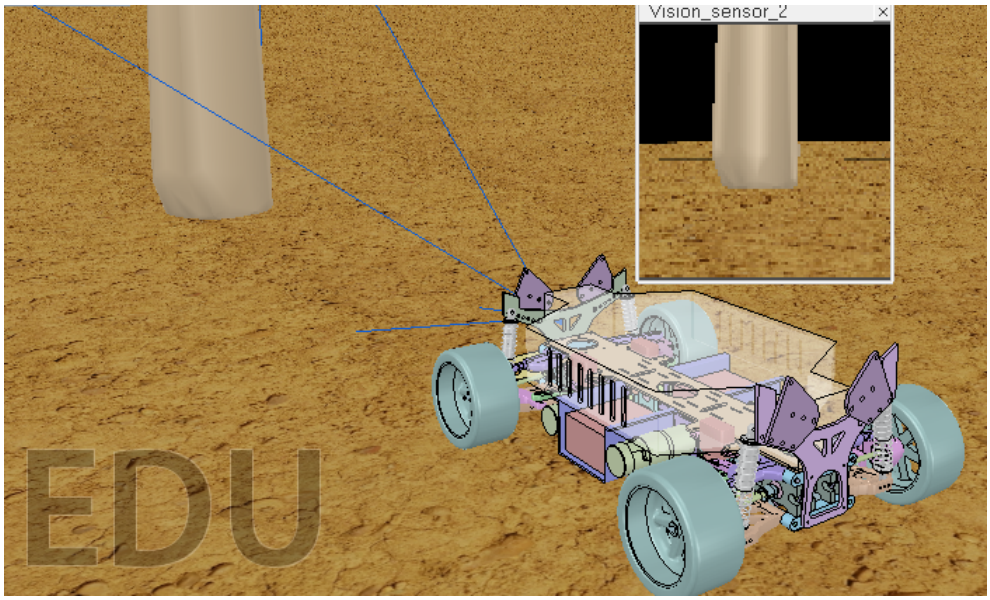
Veškerá data z proximity senzorů je možné ihned zakreslovat do grafů v rámci funcionality V-REP, což je podrobněji rozvedeno v kapitole 5.2. V případně řízení simulace Matlabem je také možnost data v každém kroku simulace exportovat a případně dále zpracovávat.

5.1.2 Vizuální senzory

Dalším typem senzoru ve V-REP jsou vizuální senzory.

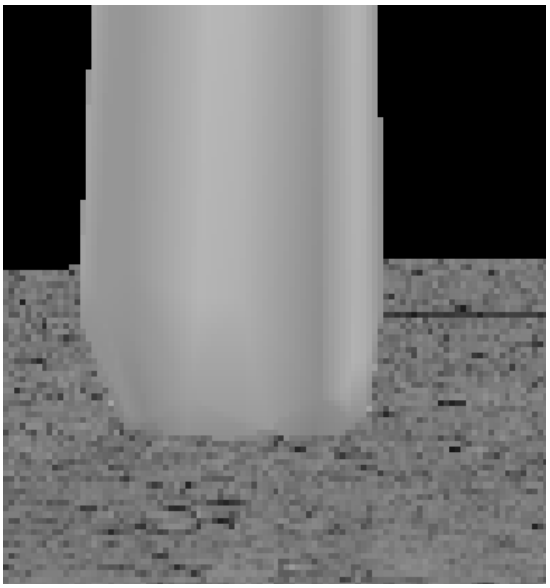
Existují dva základní typy:

- Orthographic - Jedná se o typ vizuálního senzoru, který snímá pouze oblast o daném rozlišení bez ohledu na perspektivu. Nezáleží tedy, jak moc je snímaný objekt daleko od senzoru, výsledná obraz je při změně vzdálenosti stále stejný.
- Perspective - Jedná se o typ vizuálního senzoru, v jehož výsledném obraze se již projevuje perspektiva. Z toho vyplývá, že se při změně vzdálenosti snímaného objektu od senzoru zvětšuje snímaná plocha, avšak na úkor kvality výsledného obrazu. Použití Perspective senzoru v simulaci experimentálního vozidla je zobrazeno na obrázku 5.3.



Obr. 5.3: Využití vizuálního senzoru

Výsledný obraz vizuálního senzoru lze následně zobrazit přímo v dané simulaci (5.3). V případě užití programu Matlab je možné v každém simulačním kroku exportovat obrazovou matici senzoru. Výsledná obrazová matice může být v odstínech šedi, nebo je možnost exportovat tři matice RGB. Pro každou složku barvy jedna matice. Následné zobrazení matice je již pouze otázka užití příslušné funkce v matlabu, v případě odstínů šedi např. příkaz *imshow* (platí pro Matlab verze 2013a).



Obr. 5.4: Černobílý obraz ze scény 5.3 vytvořený z obrazové matice v Matlabu

5.2 Možnosti zobrazení dat pomocí grafů [2]

Grafy v programu V-REP jsou objekty sloužící k záznamu, vizualizaci a exportování dat ze simulace.

Existují tři základní druhy grafů:

- Time graphs - slouží k zobrazení libovolné proměnné v čase
- X/Y graphs - slouži k zobrazení dvou různých proměnných v závislosti na sobě
- 3D curves - pomocí 3D curves je možné zobrazit více proměnných v závislosti na čase.

V grafech je možné zobrazit velké množství dat. V případě simulace experimentálního vozidla jsou grafy vhodné pro kontrolu činnosti proximity senzorů. Dále je možné tímto způsobem porovnávat zpoždění Matlabu při zpracování dat. Příklad zobrazení dat z proximity senzoru v čase je možné pozorovat na obrázku 5.1.

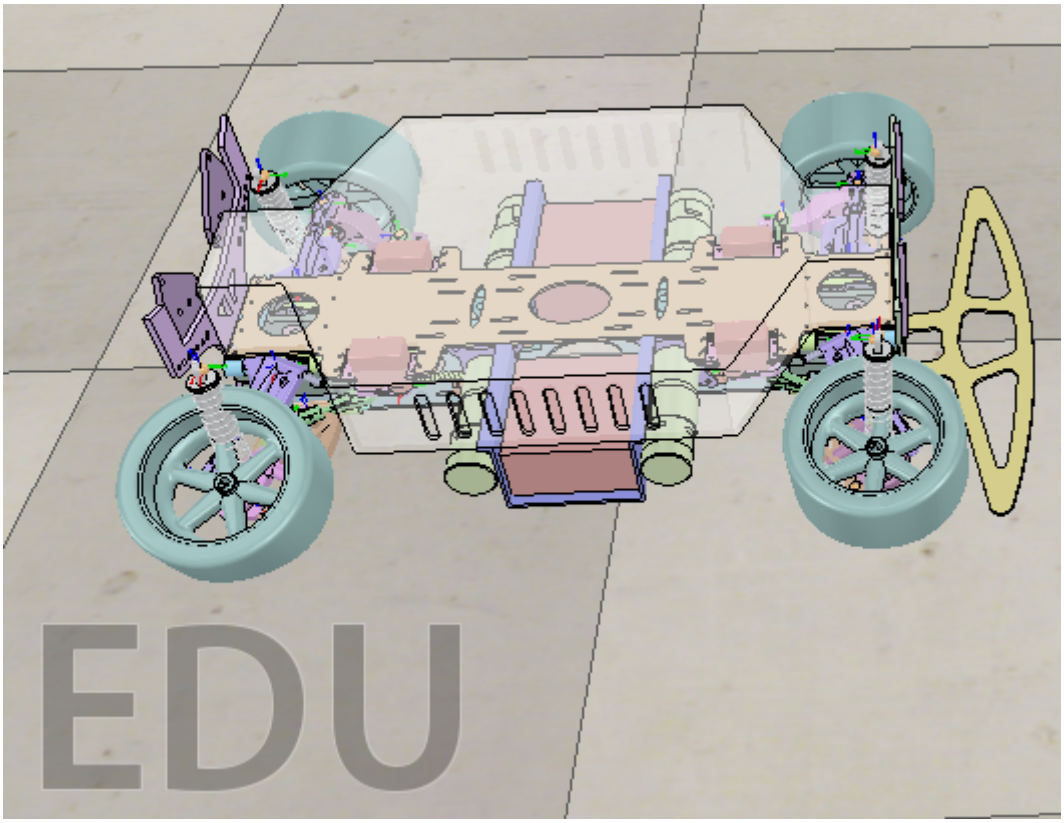
Data z grafu je možné exportovat pro další úpravy do formátu .csv („Comma Separated Values“), se kterým umí pracovat např. Matlab nebo Excel.

5.3 Užití různých řešičů

Řešiče dynamiky, se kterými V-REP pracuje, byly zmíněny v kapitole 2.7. V zásadě jsou na výběr čtyři řešiče a taktéž čtyři druhy přesnosti výpočtů pro každý řešič. V této kapitole jsou testovány kombinace různých řešičů s různých přesnostmi výpočtů, včetně následného porovnání na složitém modelu experimentálního vozidla CAR4. Vortex Dynamics Engine a Newton Dynamics Engine budou testovány pouze okrajově. Vortex Dynamics Engine je komerční verze řešiče, u které je omezeno použití v Education licenci a Newton Dynamics Engine je k aktuálnímu datu testování v Beta verzi.

Řešič - ODE, Přesnost - Very fast

Již při spuštění simulace lze pozorovat v klidovém stavu, že vazby začínají kmitat. Při rozjezdu se kola náhodně otáčejí, při pokusu o zatočení přestává fungovat omezení vazeb a kola na jedné straně vozu se převrací, viz obrázek 5.5. Přesnost Very fast tedy není vhodná v našem případě, lze ji tedy doporučit u velmi jednoduchých modelů, kde není přesnost řešení stěžejní.



Obr. 5.5: Nefunkčnost vazeb, řešič: ODE, přesnost: Very fast

Řešič - ODE, Přesnost - Fast

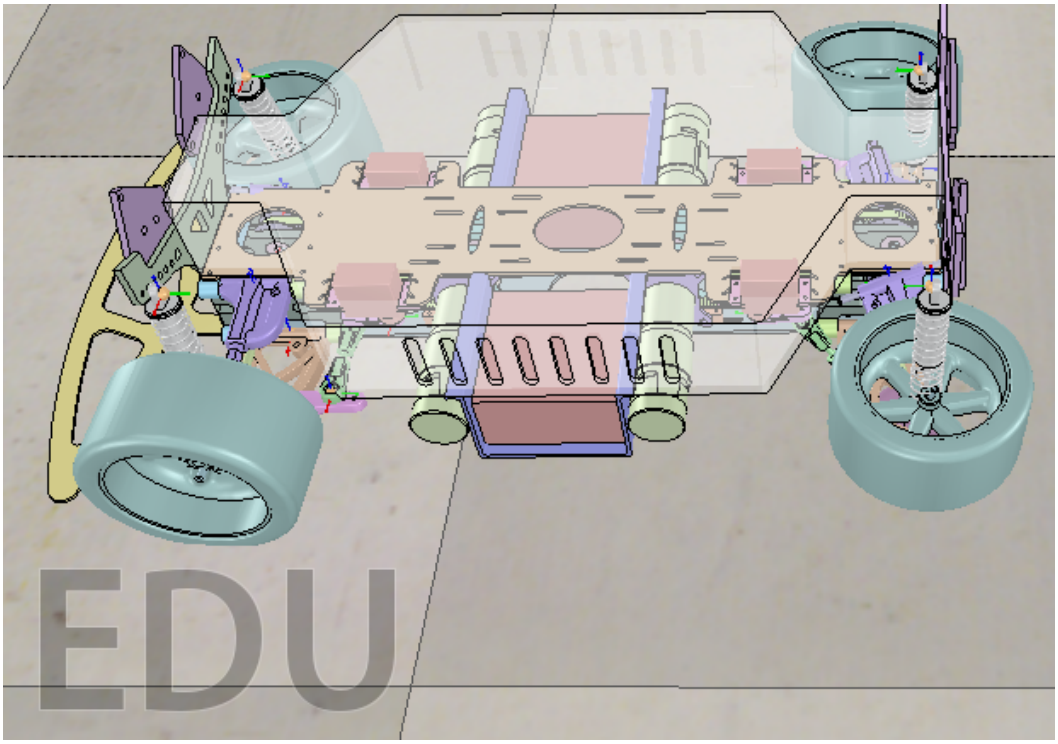
V klidovém stavu i při rozjezdu je model v pořádku. Při velkých rychlostech lze pozorovat samovolné natáčení kol. To je způsobeno přestávajícím fungováním omezení u vazeb. Oproti předchozímu režimu však již nedochází k převrácení kol.

Řešič - ODE, Přesnost - Accurate, Very Accurate

Při spuštění simulace se model chová stabilně. Obdobně v zatáčkách. Výpočet je dostatečně rychlý pro správný výpočet omezení, která jsou definována ve vazbách modelu. Při zvýšení přesnosti na Very accurate je situace obdobná, avšak zvyšuje se tímto zbytečně náročnost výpočtu v simulaci. Lze tedy vyvodit závěr, že řešič ODE s přesností Accurate je dostatečný pro správný chod simulace.

Řešič - Bullet, Přesnost - Very fast

Při použití řešiče Bullet je již situace podstatně horší. V případě přesnosti Very fast výpočet propojení vazeb a součástí nefunguje správně. Vazby mezi koly a rámem vozidla téměř nejsou zřetelné, neboť celá statická část vozu není schopná udržet na nápravách a spadne na pracovní plochu. Model není schopen rozjetí ani zatočení. Pro řesič Bullet rozhodne nelze přesnost Very fast doporučit.



Obr. 5.6: Nefunkčnost vazeb, řešič: Bullet, přesnost: Very fast

Řešič - Bullet, Přesnost - Fast, Accurate

Při přesnosti Fast a Accurate u řešiče Bullet je již vozidlo schopné částečného rozjetí, avšak omezení vazeb funguje velmi špatně. Při větších rychlostech a ostřejším zatáčení přestávají fungovat úplně. Tyto přesnosti pro řešič Bullet nelze v případě modelu experimentální vozidla doporučit.

Řešič - Bullet, Přesnost - Very accurate

S přesností Very accurate je již situace a poznání lepší. Vozidlo již reaguje správně při rozjezdu i při zatočení. Při vyšší rychlosti a prudším zatočení ale také přestává fungovat omezení vazeb a model se částečně rozpadá. S jistými výhradami je řešič Bullet při přesnosti Very accurate použitelný pro simulaci experimentálního vozidla, avšak i tak jej nelze zcela doporučit.

Řešič - Vortex, Přesnost - Accurate

Řešič Vortex je testován pouze pro přesnost Accurate, z důvodu limitované činnosti řešiče na 20 sekund. Vazby použité v modelu však s tímto řešičem nepracují správně, místo rozjezdu vozidla dojde pouze k převrácení na střechu, z čehož lze odvodit, že tento typ řešiče používá jiné přístupy pro řešení dynamiky, než řešiče v předchozích případech. Není možné použít pro model experimentálního vozidla projektu CAR4.

Řešič - Newton, Přesnost - Accurate

Řešič Newton je testován také pouze pro přesnost Accurate, neboť jeho současný stav implementace do V-REP je v Beta verzi. Chování je velmi podobné jako u řešiče Vortex, model nereaguje správně na zrychlení, po čase začne samovolně létat vzduchem. Není použitelný pro model experimentálního vozidla projektu CAR4.

Z výše uvedeného testování vyplývá, že řešič ODE se ukázal jako nejvhodnější pro simulaci modelu experimentálního vozidla projektu CAR4. Vykazuje nejpřesnější a nejméně náročné výpočty. Taktéž autor webové stránky [3] (tutoriál pro využití Matlabu společně s V-REP) doporučuje používat jako výchozí řešič pro všechny simulace ODE.

5.4 Rychlost simulace a komunikace s Matlabem

5.4.1 Tvorba real-time komunikace s Matlaben

U simulace ve V-REP je možnost nastavit výpočetní dobu jednoho výpočetního cyklu. Výchozí hodnota je nastavena na periodu T = 50 ms. Periodu je možné změnit a může nabývat hodnot

T = 10 ms, 25 ms, 50ms, 100 ms, 200 ms.

V případě, že je potřeba simulaci spustit v reálném čase, je zde možnost využití režimu real-time. Je však podstatné správně volit periodu jednoho kroku simulace. V případě, že je hodnota periody nastavena na nižší hodnotu, než jsou výkonnostní možnosti hardwaru, na které je simulace spuštěna, může nastat nežádoucí prodloužení periody jednoho simulačního cyklu. Např. simulace poběží s periodou
T = 50 ms (frekvence f = 20 Hz), avšak skutečný čas potřebný pro výpočet bude
T = 62 ms (frekvence f = 16,2 Hz).

V Matlabu oproti V-REP však výpočty probíhají co nejrychleji. Aby nedošlo k situaci, kdy jsou data zasílána do V-REP vícekrát za jeden simulační krok, je nutné využít funkci *simxSynchronous*, která zajišťuje synchronizaci mezi oběma programy.

```
vrep.simxSynchronous(id,true)
```

Aby synchronizace probíhala správně, je taktéž nutné použít funkci *simxSynchronousTrigger*, která určuje, kdy má začít odeslání příkazů pro následující periodu simulace, viz obrázek 4.2. Tuto funkci je nutné umístit v rámci cyklu, aby bylo dosaženo opakovaného vykonání příkazů pro každý simulační krok.

```
vrep.simxSynchronousTrigger(id)
```

5.4.2 Testování Real-time řízení

Poté, co proběhla synchronizace mezi Matlabem s V-REP je nutné ověřit, zda při real-time řízení perioda simulace (případně frekvence) opravdu nabývá takových hodnot, jaké byly nastaveny v parametrech simulace. Toho je možno dosáhnout změřením času potřebného pro vykonání jednoho cyklu příkazů pro daný simulační krok v Matlabu. Pro měření délky trvání cyklu byly použity Matlab příkazy *Tic* a *Toc*, určené právě pro tento účel. Měření probíhalo pro jednoduchý i složitý model a zároveň i pro jednotlivá nastavení periody simulace.

Specifikace měření:

Pro každé měření bylo užito 500 hodnot (500 cyklů simulace), celková doba simulace se tedy liší s ohledem na užitém výpočetním kroku. Simulace probíhala ve výchozím prostředí ve V-REP bez užití senzorů, grafů a pouze pro výchozí kameru. Modely byly v rámci měření řízeny v dopředném pohybu po kružnici. Dle výsledků v kapitole 5.3 byl pro měření užit řešič ODE s přesností Accurate.

Specifikace hardwaru užitého při měření:

Procesor:	Intel Core i5-2500 CPU @ 3.30 GHz 3.60 GHz
RAM:	4,00 GB
Grafická karta:	NVIDIA Quatro 400
Harddisk:	Seagate ST500DM002
Operační systém:	Windows 7 Professional, 64 bit

Výsledky měření pro složitý model	Měření 1.1	Měření 1.2	Měření 1.3	Měření 1.4	Měření 1.5
Nastavená perioda jednoho cyklu T [ms]	10,0	25,0	50,0	100,0	200,0
Skutečná průměrná perioda jednoho cyklu T [ms]	36,0	38,1	60,1	102,9	200,0
Nastavená frekvence simulace f [Hz]	100,0	40,0	20,0	10,0	5,0
Skutečná frekvence simulace f [Hz]	27,5	26,2	16,6	9,7	5,0
Nastavená doba simulace t [s]	5,0	12,5	25,0	50,0	100,0
Skutečná doba simulace t [s]	18,0	19,1	30,1	51,5	100,0

Výsledky měření pro jednoduchý model	Měření 2.1	Měření 2.2	Měření 2.3	Měření 2.4	Měření 2.5
Nastavená perioda jednoho cyklu T [ms]	10,0	25,0	50,0	100,0	200,0
Skutečná průměrná perioda jednoho cyklu T [ms]	18,9	25,5	50	100,0	200,0
Nastavená frekvence simulace f [Hz]	100,0	40,0	20,0	10,0	5,0
Skutečná frekvence simulace f [Hz]	52,9	39,4	20	10,0	5,0
Nastavená doba simulace t [s]	5,0	12,5	25,0	50,0	100,0
Skutečná doba simulace t [s]	9,5	12,7	25	50,0	100,0

Z výsledků vyplývá, že na Real-time řízení má náročnost simulace velký vliv. U jednoduššího modelu probíhala simulace opravdu v reálném čase pouze pro Měření č. 2.3, Měření č. 2.4 a Měření č. 2.5. U Měření č. 2.2 již lze pozorovat mírnou odchylku mezi reálným časem a časem simulace. U Měření č. 2.1 je tento rozdíl ještě více zřetelný. U složitého modelu byla situace podstatně horší. Pouze u Měření č. 1.5 se reálný čas téměř nelišil od simulačního, ale u všech ostatních měření se již lišil poměrně podstatně. Dále je možné dle očekávání pozorovat, že výpočetní náročnost simulace roste s rostoucí použitou vzorkovací frekvencí. U frekvencí f = 100 Hz a f = 40 Hz se simulace zpožďovala v rámci obou měření.

Z výše uvedeného je možné říci, že testovací hardware je vhodný pouze pro real-time řízení jednoduššího modelu, a to pro frekvenci f = 20 Hz nebo nižší. V případě náročnějších real-time simulací je tedy nutné zvážit použití hardwaru s výkonnějšími komponenty

V jistých případech může docházet i k několikasekundovému zpoždění v rámci jednoho kroku. Například při odesílání obrazové matice do Matlabu s příliš velkým rozlišením dochází i k několikanásobnému překročení nastavené periody. Je tedy třeba tyto vlivy pokud možno omezit před startem simulace.

5.5 Jizda po předem určené dráze

K velmi užitečných objektům, které lze ve V-REP vytvořit, patří objekt Path. Ve speciální nabídce Toogle Path Edit Mode lze následně definovat otevřenou nebo uzavřenou trasu ve scéně. Toho lze například využít pro naprogramovaní modelu experimentálního vozidla pro jízdu po této trase.

Nejprve je nutné definovat trasu. To je možné pomocí výše zmíněného Toogle Path Edit Mode, kde je možné jednoduše, pomocí určení prostorových souřadnic nebo tahem myši, definovat body trasy ve scéně, které se následně automaticky propojí pomocí interpolace dle kriterií, která nastavíme. Následně je třeba vytvořit pomocné objekty Dummy, pro algoritmizaci pojmenované jako *pathfollower* a *pathfollowertarget*. Jeden bude umístěn k vozidlu, druhý k objektu Path. Algoritmus je vysvětlen v následujícím odstavci.

Postup algoritmizace:

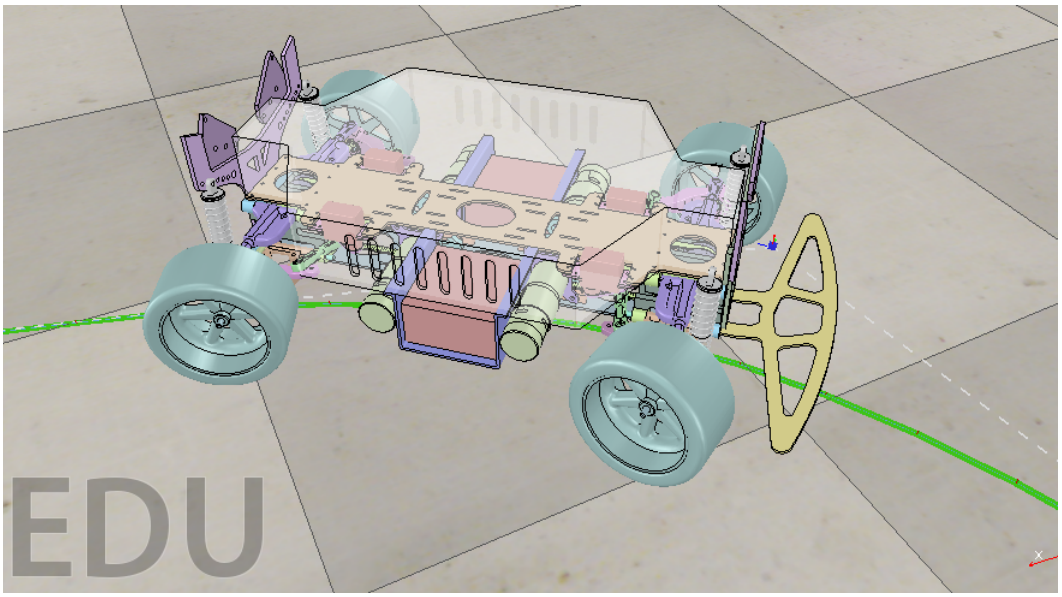
```
% Nalezení nejbližšího bodu na trase, který bude následován
    m = simGetObjectMatrix(ram,-1)
    inclination = m[5]
    pf = simGetObjectPosition(path_follower,-1)
    m3 = simGetObjectMatrix(path,-1)
    m3 = simGetInvertedMatrix(m3)
    ppf = simMultiplyVector(m3,pf)
    l = simGetClosestPositionOnPath(path,ppf)
    pp = simGetPositionOnPath(path,l)
    simSetObjectPosition(path_follower_target,-1,pp)
    v = simGetObjectPosition(path_follower_target,path_follower)

% Výpočet požadované změny úhlu
    correction=v[2]*0.5
    if (math.abs(correction)>0.5) then
        correction=correction*0.5/math.abs(correction)
    end

% Aplikace vypočteného úhlu na výchozí úhel
    inclination=inclination-correction
```

Popis algoritmu:

Pomocí API funkce *simGetObjectMatrix* je třeba určit umístění rámu vozidla v prostoru. Dále určíme výchozí úhel pro výpočet (proměnná *inclination*).
Poté je nalezen nejbližšího bod (*m3*) k objektu „pathFollower“ dummy umístěný na objektu *Path*, který bude následován. Proměnná *ppf* je pozice objektu dummy „pathFollower“ relativně k souřadnicovému systému objektu *Path*. Proměnná *pp* nejbližší pozice k proměnné *ppf*, která je na objektu *Path*. Poté je přenesena pozice objektu dummy „pathFollowerTarget“ na *pp* pozici. Proměnná *v* je pozice objektu dummy „pathFollowerTarget“ relativně k souřadnicovému systému objektu dummy „pathFollower“. Nakonec je vypočítána korekce úhlu zatáčení a tato korekce je užita pro výchozí úhel.



Obr. 5.7: Jízda po předem vytyčené trase.

5.6 Hledání vhodné trasy (Path Planning)

Program V-REP nabízí také funkci pro hledání ideální trasy. Používá k tomu výpočetní modul Path planning a objekt Path, který je následným výpočtem upraven do podoby hledané trasy. Jedná se poměrně jednoduché rozhraní, kde se pomoci umístění objetu Dummy definuje počáteční a cílový bod na trase. Dále je potřeba využít objektu, který bude sloužit jako transportní medium na vypočtené trase. Počáteční objekt Dummy musí být umístěn jako potomek vzhledem k objektu Path a spolu s transportním médiem musí ležet na stejných absolutních souřadnicích v prostoru. Nakonec je třeba určit velikost a umístění plochy, na které bude hledání optimální trasy probíhat a v jakých souřadnicích bude trasa vyhledána. Poté je již proveden samotný výpočet. U výpočtu lze také nastavit minimální vzdálenost od kolizních objektů, avšak tato funkce fungovala velmi nedokonale a mnohdy trasa procházela skrz kolizní objekty.

Nástroj Path planning se ukázal jako funkční, ale poměrně nedokonalý. U složitějších modelů nebylo možné nalézt žádnou trasu. Také u objektů větších než 100 mm bylo hledání velmi nedokonalé. Pouze u malých a jednoduchých objektů probíhal výpočet dle očekávání. Pokus o využití ve zjednodušeného modelu experimentálního vozidla nebyl úspěšný, vzhledem k jeho velikosti a složitosti.

V nejnovější verze programu V-REP je možné integrovat plugin OMPL, který je určen právě pro výpočty ideální trasy Path planning a výpočty pohybu modelů v prostoru Motion planning. Je také nově oficiálně doporučován samotnými vývojáři k využívání k těmto účelům, namísto integrovaných výpočetních modulů užitých v rámci této kapitoly.

6 ZÁVĚR

V rámci této práce se software V-REP ukázal jako velmi solidní nástroj pro využití v robotice, zejména v oblasti simulace dynamiky, modelování vazeb a také jako skvělý nástroj pro vizualizaci v součinnosti s programem Matlab.

Nejprve bylo nutné prostudovat konstrukci a CAD model experimentálního vozidla projektu CAR4. Poté byly vytvořeny dva modely tohoto vozidla. Jednodušší model byl určen k poznávání mechanik, modelovacích postupů a testovacích postupů v rámci programu V-REP. V dalším kroku byla snaha o vytvoření složitějšího modelu, co možná nejpřesněji napodobujícího vazby a dynamiku jako v případě reálného modelu experimentálního vozidla. Tohoto cíle bylo z větší části dosaženo. Bylo však nutné přijít s ústupky v oblasti tvarové reálnosti jednotlivých součástí. Současný běžný hardware není na takové úrovní, aby bylo možné vytvářet real-time simulace dynamicky tvarově složitých modelů. Další limitace plynoucí z nutnosti tvarového zjednodušení modelu vozidla byly také v oblasti užití vazeb, zejména při modelování přenosu momentu.

V další fázi práce byly ukázány možnosti v oblasti řízení modelů. Toho je možno dosáhnout přímo s využitím vnitřních skriptovacích možností programu V-REP, kde je využita struktura programovacího jazyku Lua. Zajímavější volbou pro řízení modelů se však ukázalo využití programu Matlab. Díky již postavené komunikaci a implementaci funkcí, které V-REP využívá, se Matlab ukázal jako výborný nástroj, jak pro již zmíněné řízení modelů, tak zejména jako prostředek pro získávání a zpracování dat ze simulací. Přes drobná omezení v oblasti funkcí ovlivňujících uživatelné prostředí a vnitřní nastavení V-REPu je možné v Matlabu využít téměř všechny funkce, jako v případě užití vestavěných skriptů.

Další výhodou V-REP je také real-time řízení modelů v rámci simulace, např. přes již zmíněný Matlab. Toho je možné dosáhnout s maximální frekvencí f = 100 Hz, ale s ohledem na náročnost modelu je zde nutné počítat s užitím dostatečně výkonného hardware. Na hardware použitém pro simulace v rámci práce se podařilo řídit zjednodušený model v real-time režimu pouze při frekvenci f = 20 Hz, složitý model pak s frekvencí pouhých f = 5 Hz. Tento výsledek se již ukazuje pro real-time řízení jako téměř nepoužitelný.

Ve V-REP je také možnost tvorby prvku Path, který, díky širokým možnostem naprogramování, je možné skvěle využít při řízení modelů v prostoru. Další užitečné nástroje jsou simulace proximity senzorů a vizual senzorů, které se v kombinací s

real-time řízením a zpracováním dat ukázaly jako výborný nastroj pro řízení a orientaci modelů v prostoru.

Vše výše zmíněné naznačuje potenciál programu V-REP, který by bylo možné využít například pro mapování neznámého prostoru, případně algoritmů vyvinutých pro tento účel. Další velmi zajímavá možnost je využití např. Robotic Toolboxu, určený pro program Matlab, k řízení robotů a manipulátorů a následnou vizualizaci ve V-REP, podobně jako je tomu v [3]. Jako přínosné se zdá také prozkoumání pluginu OMPL. Tento nástroj je podporovaný a doporučovaný v nové verzi programu V-REP a je určen pro efektivní Motion a Path planning.

LITERATURA

[1] JASANSKÝ, Michal. *Návrh dynamických modelů pro řízení trakce experimentálního vozidla.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2010. 123 s. Vedoucí diplomové práce Ing.Robert Grepl, Ph.D.

[2] COPPELIA ROBOTICS *V-REP User Manual* [online]. 2016, [cit. 13. 5. 2016]. Dostupné z URL: <`http://www.coppeliarobotics.com/helpFiles/`>.

[3] DETRY, Renauld, Peter Corke, Coppelia Robotics *TRS: An Open-source Recipe for Teaching/Learning Robotics with a Simulator.* [online]. 2016, [cit. 13. 5. 2016]. Dostupné z URL: <`http://ulgrobotics.github.io/trs`>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

API	„Application Programming Interface“
.csv	„Comma Separated Values“
ROS	„Robot Operating system“
SS	„Souřadnicový systém“ - Coordinate system
.stl	„STereoLithography“
URDF	„Unified Robot Description Format“
V-REP	„Virtual Robot Experimentation Platform„ - Program pro simulaci robotických modelů
f	„Frekvence [Hz - Hertz]“
T	„Perioda [s - sekunda, ms - milisenda]“
t	„Čas [s - sekunda]“

A PŘÍLOHA

Obsah přiloženého CD:

- Jednoduchý model vozidla CAR4 - řízený užitím V-REP child scriptem pomocí vstupů z klávesnice.
- Složitý model vozidla CAR4 - řízený užitím V-REP child scriptem pomocí vstupů z klávesnice.
- Složitý model vozidla CAR4 se senzory - řízený užitím Matlab M-file pomocí joysticku.
- Scénu se složitým modelem - model vykovává jízdu po předem určené trase.
- Soubor Auto.m - soubor nutný pro řízení pomocí Matlabu.
- Soubory remApi.m, remoteApi.dll a remoteApiProto.m - soubory nutné pro komunikaci mezi V-REP a Matlab.
- Soubory vrchk.m a cleanup-vrep.m - soubory nutné pro funkčnost Auto.m.