Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Disertační práce

Moderní řízení pohybových stavů mechanické soustavy průmyslového robota prostřednictvím elektromechanických akčních členů

Modern motion control methods of an industrial robot mechanical system through electromechanical actuators

Autor: **Ing. Jiří Zátopek**
Obor: Inženýrská informatika
Školitel: doc. RNDr. Ing. Zdeněk Úředníček, CSc.

Zlín, 2022

Na tomto místě bych rád poděkoval všem, kteří mě v průběhu celého studia podporovali, motivovali a byli mi oporou. V první řadě paří mé díky rodině za nehynoucí trpělivost, velkorysost a pochopení, přátelům za věčný optimizmus a upřímnost, kolegům za otevřenost a obětavost, a v neposlední řadě mému vedoucímu Zdeňkovi Úředníčkovi za cenné rady, konzultace a velice vstřícný a přátelský přístup v celém průběhu studia.

„Kdo víno má a nepije, kdo hrozny má a nejí je, kdo ženu má a nelíbá, kdo zábavě se vyhýbá, na toho vemte bič a hůl, to není člověk, to je vůl.“

Jan Werich

ABSTRAKT

Tato dizertační práce se primárně zabývá porovnáním moderních neautonomních metod řízení pohybu, se standardní, léty prověřenou autonomní regulační strukturou, a to na reálném mechatronickém/robotickém systému s dynamickými projevy odpovídajícími chování sériového průmyslového robota antropomorfního typu. V první části práce jsou rozebírány možné typové struktury systému, je vytvořen kompletní konstrukční CAD model a vybráno elektrotechnické vybavení. Dále je popisována výroba první verze prototypu a představeno finální řešení, které zahrnuje vlastní výrobu, sestavení a zprovoznění celého systému. Následující pasáž je věnována softwarové aplikaci, která obsahuje ukázku její struktury s popisem tříd, funkcí, způsobem vzájemné komunikace mezi řídicí a výkonovou částí zařízení a grafickým uživatelským rozhraním. Nezanedbatelná část práce je věnována zpracování obrazu z kamery, kde je detailně popsán celý postup vyhodnocení snímku včetně ukázek zdrojového kódu. Konečná fáze kamerového vyhodnocení využívá komplexní kinematické transformace, kterým je věnována další kapitola. Tyto transformace používají k odvození obecné matice homogenní transformace mimo jiného i CAD model, jehož části jsou využívány v podstatě v celé práci, a výsledky jsou aplikované jak na obrazové vyhodnocení, tak při návrhu zákona řízení. Před samotným odvozením regulátorů je sestaven matematický a fyzikální model, z nichž první zmiňovaný slouží k návrhu řízení a pro základní analýzu chování, druhý uvedený zase k simulacím reálného chování systému, ověření správnosti odvození matematického modelu, ladění regulátoru, ale i např. k výběru akčních členů. Oba modely jsou součástí největší kapitoly, zabývající se řízením pohybu. Tato kapitola popisuje regulaci nakloněné roviny a kuličky, od nastavení proudové smyčky, přes návrh kaskádní P(I)(D) regulace, až po odvození a implementaci jedné z moderních metod řízení pohybu - výpočtu točivých momentů. Všechny naměřené regulační pochody jsou statisticky vyhodnoceny a průběhy vykresleny v přehledných grafech. V poslední části práce je stanoveno celkem 5 kritérií posuzujících kvalitu regulace, na jejichž základě jsou porovnány oba přístupy k řízení pohybu. Zvláštní důraz je po celou dobu řešení kladen na ověření všech dílčích cílů na reálném robotickém systému.

SUMMARY

This doctoral thesis primarily focuses on comparing modern non-autonomous motion control methods with a standard, years-proven autonomous control structure on a real-build mechatronic/robotic system with dynamics appropriate to the industrial serial robot behaviour of an anthropomorphic type. In the first part of the work, the possible kinematic structures of the system are analyzed, a complete structural CAD model is created, and the electrical equipment is selected. Next, the prototype fabrication is introduced, and the final solution is presented, which includes the construction, assembly and commissioning of the entire system. The following section is devoted to the software application, which includes a presentation of its structure with a description of classes and functions, the method of mutual communication between the control and power parts of the device, and a graphical user interface. A comprehensive part of the work contains camera image processing, where the entire image evaluation process is described in detail, including source code samples. Camera evaluation uses complex kinematic transformations covered in the next chapter. These transformations use, among others, a CAD model to derive the general transformation matrix, parts of which are used essentially throughout the work, and the results are applied both to image evaluation and to the design of the control law. Before the controllers' derivation, a mathematical and physical model was created. The first mentioned is for the design of the controllers and the fundamental demeanour analysis. The second is for behaviour simulations of the existing system, the mathematical model correctness verification, tuning of the controller, and, for example, the selection of actuators. Both models are part of the most significant chapter dealing with motion control. This chapter describes the inclined plane and ball position regulation, from the current loop setup, through the P(I)(D) cascade regulation design, to the derivation and implementation of one of the modern motion control methods - the computed torques. All measured control processes are statistically evaluated and presented in clear graphs. In the last part of the thesis, five criteria assessing the quality of regulation are established, based on which both motion control approaches are compared. Particular attention is placed on verifying all sub-goals on a real-build robotic system.

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM UKÁZEK ZDROJOVÝCH KÓDŮ

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CAD	Computer Aided Design
STL	STereoLithography
MATLAB	MATrix LABoratory
VRML	Virtual Reality Modeling Language
PMSM	Permanent Magnet Synchronous Motor
CCD	Charce-Coupled Device
CMOS	Complementary Metal-Oxide Semiconductor
OpenCV	Open Source Computer Vision Library
GUI	Graphical User Interface
IDE	Integrated Development Environment
DOF	Degree Of Freedom
UDP	User Datagram Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
CNC	Computer Numeric Control
ABS	Akrylonitril-Butadien-Styren
PETG	Glykolem modifikovaný Polyethylene Terephthalate
SCI	Serial Communication Interface
FPS	Frames Per Second
HDMI	High-Definition Multi-media Interface
USB	Universal Serial Bus
EtherCAT	Ethernet for Control Automation Technology
MAC	Media Access Control
VDC	Voltage Direct Current
VAC	Voltage Alternating Current
BGR	Blue Green Red
LAB	Luminance A B
HSV	Hue Saturation Value
RPi	Raspberry Pi
DFT	Discrete Fourier Transform
GSS	Globální Souřadný Systém
LSS	Lokální Souřadný Systém

SS	Souřadný Systém / Souřadná Soustava
D-H	Denavit-Hartenberg
MIMO	Multi Input MultiOutput
RK	Regulace Kaskádní
RVM	Regulace Výpočtem Momentů

1 ÚVOD

Řízení pohybu tuhých těles vázaných mezi sebou je ve své podstatě vždy nelineární záležitostí - ať už kvůli tzv. tvrdým nelinearitám (tření, vůle, saturace), nebo kvůli odstředivé/dostředivé a Coriolisově síle apod. U složitějších struktur, zejména rychle se měnících ve stavovém prostoru, není vždy možné provést linearizaci, která by umožnila určit stabilitu a požadovanou efektivitu regulace. Adaptivní řízení také nemá vždy uspokojivé výsledky a může představovat problémy s robustností. Řešení spočívá v získání co možná nejpřesnějšího nelineárního modelu. S tímto modelem je možné pomocí různých účinných nástrojů dále pracovat a pro návrh nelineárního zákona řízení je nutné jej znát.

Samotné řízení bude probíhat na navržené a reálně sestavené sériové robotické struktuře se čtyřmi zobecněnými stupni volnosti, z nichž dva jsou přímo řiditelné. Popisovaný model sestává z nakloněné roviny řízené dvěma sériově uloženými elektromotory (v podstatě kardanovo uložení). Řízeným objektem je kulička, která se po této rovině odvaluje (ne vždy ideálně) a rozšiřuje model o další dva nepřímo řízené zobecněné stupně volnosti (posuv kuličky v osách nakloněné roviny). Uspořádání modelu bude navrženo s důrazem na nezanedbatelný vliv kuličky, „měkkých“ nelinearit a vzájemné působení jednotlivých aktuátorů v celkovém chování systému při dynamických pohybech.

Obvyklý a nejjednodušší způsob řízení pohybových stavů manipulátorů a orientačních systémů robotů je použití kaskádních PID regulátorů na autonomní řízení momentu, rychlosti a polohy jednotlivých kloubů. Pří návrhu tohoto způsobu řízení jsou zanedbávány změny koeficientů matice setrvačnosti vlivem pohybu soustavy, a vzájemné silové působení mezi jednotlivými částmi manipulátoru a zápěstí. Ty potom musí autonomní regulátory kompenzovat svou robustností vůči změnám parametrů soustav, resp. vůči poruchám. Takto navržený systém regulující pohybové stavy manipulátorů většinou nezvládá přesné sledování trajektorií při větších rychlostech pohybu, resp. vede k výkonovému naddimenzování akčních členů. To je neefektivní a pro kvalitní řízení pohybu s rychle se měnícími trajektoriemi vede i k předimenzování mechanických konstrukcí.

Jedním z řešení uvedeného problému jsou metody regulace založené na znalosti nelineárního dynamického modelu řízených soustav - metody pro řízení

systémů s více vstupy a více výstupy. Také jsou k dispozici pokročilé metody linearizace, které se zcela liší od lineární aproximace dynamiky výpočtem Jacobiho matice ve stacionárním (pracovním) bodě.

Přínosem této práce je tedy možnost využití odlišného způsobu řízení, což by mohlo vést k optimalizovanému návrhu robotických struktur, ať už výkonově, rozměrově, materiálově, a tedy i cenově.

2 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

Se stále rostoucím výpočetním výkonem počítačů a jejich klesající cenou, je možné použít pro řízení pohybu v reálném čase metody, které byly dříve aplikovatelné pouze teoreticky, simulačně, nebo s nutností offline výpočtu alespoň části zákona řízení. V současné době je s využitím hardwarových prostředků v řádech tisícikorun reálné implementovat v jednom minipočítači velmi výpočetně složitý algoritmus řízení, komunikační rozhraní, zpracování obrazu z kamery, grafickou uživatelskou aplikaci s online parametrizací celého systému, a to vše vykonávat rádoby současně a s opakovací frekvencí regulace nad 1 kHz. Takováto cenově dostupná řešení nejsou sice aplikovatelná v průmyslové sféře, protože nejen z důvodu absence systému reálného času nedosahují požadované spolehlivosti chodu, ale na ověření použitelnosti a základní funkčnosti navrhovaného řešení jsou více než dostatečná. Následné přepracování do průmyslově vhodného konceptu už poté není tak složité, a díky hotové analýzy a syntéze řešeného problému je celou výslednou sestavu jednodušší parametrizovat. Tato práce si klade za cíl vytvořit právě takovýto spojující článek mezi teoretickým návrhem, podpořeným simulačními experimenty, a praktickou realizací, použitelnou v reálných podmínkách.

2.1 Řízení pohybu

Metod řízení pohybu je nepřeberné množství, a speciálně pro řešený systém „Ball & Plate“ - Kulička na nakloněné rovině, jsou publikovány mnohé z nich. Mezi nejznámější a na tomto systému ověřené patří např. PID a fuzzy regulace [47, 32, 13, 41], sliding-mode (regulace v klouzavém režimu) [47, 29, 56, 32, 9, 41, 48],

computed-torque (výpočet momentů) [59], LQR (lineární kvadratický regulátor) [32], adaptivní a prediktivní metody regulace [70, 29, 51], zpětnovazební linearizace [10], neuronové sítě [62, 52] a mnoho dalších. Většina publikací také zahrnuje vzájemné porovnání různých řídicích strategií [14, 46, 59, 2], z kterých je možné vyvodit vhodnost jejich použití pro danou mechanickou strukturu. Jako aktuátory se pro běžné testovací aplikace využívají stejnosměrné a krokové motory, pro precizní řízení pak PMSM, ale jsou vyvinuty i různé speciální polohovací mechanismy v magnetickém závěsu [34], což je v podstatě varianta lineárních PMSM.

2.2 Snímání stavových veličin

Snímání stavových veličin aktuátorů je řešeno ve všech případech senzory úhlu natočení, které jsou většinou již součástí samotného pohonu (Hallovy sondy, enkodéry, resolvery, a jejich kombinace). Ke zjištění polohy objektu bez přímé kinematické vazby se systémem (pozice kuličky na nakloněné rovině) je však nutné využít zcela jiných metod. Dostupné zdroje představují zpravidla 3 možné způsoby, kterými řeší tento problém.

Mezi první a ve své podstatě nejprimitivnější metodu patří kontaktní snímání pomocí odporové/kapacitní dotykové vrstvy [32, 19, 9, 2, 41]. Tento způsob má mnohá omezení a úskalí, ale dosahuje vysokých vzorkovacích frekvencí a celá dotyková podložka včetně elektroniky s vyhodnocením lze pořídit jako hotové řešení, čehož je také v drtivé většině případů využíváno.

Druhou, v současnosti nejperspektivnější metodou, je optická detekce řízeného objektu pomocí kamery se softwarem vyhodnocujícím obrazovou informaci. Ve většině případů je kamera umístěna stacionárně a nepohybuje se spolu s náklonem roviny [71, 56, 68, 11, 24, 13, 46, 27]. Zde je však problém s přesnou rekonstrukcí polohy kuličky, když je kamera, byť mírně, vychýlená mimo střed. Také se v publikacích vůbec neřeší závislost náklonu roviny ve vztahu ke skutečné poloze kuličky, měřené pevně umístěnou kamerou. Uvažuje se pohyb v blízkém okolí horizontální pozice roviny a nepřesnost vznikající náklonem, která není zanedbatelná ani pro ideální uložení kamery (autorova publikace [5]), je zanedbávána. Tento problém částečně řeší umístění kamery do závěsu spojeného s nakloněnou

rovinou [47, 76], čímž se ale výrazně zvětšuje setrvačnost celé pohyblivé části systému (o kabelovém uložení a jiných negativních aspektech ani nemluvě). Co se týče popisu metod zpracování obrazu pro detekci a následné sledování kuličky, zabývají se jimi pouze některé publikace a jen okrajově [76, 52, 27].

Poslední metodou zjištění polohy kuličky na nakloněné rovině je její rekonstrukce pomocí lineárního/nelineárního pozorovatele stavu [45]. Zde je možné na základě snímaného proudu/napětí a přesné znalosti dynamického modelu detekovat polohu řízeného objektu, ovšem pouze za předpokladu, že má změna jeho pozice významný vliv na dynamické chování celé sestavy. Použití této metody je proto dosti omezené, výsledky jsou vždy zatíženy jistou mírou neurčitosti a přesnost rekonstruované veličiny se mění v průběhu regulace podle vlivu řízeného objektu na systém.

2.3 Simulace vs realita

Výsledné vyhodnocení řešeného problému je pro použití v reálných podmínkách nutné ověřit na reálně postaveném systému. Ověření na matematickém/fyzikálním modelu není dostačující a lze jej považovat pouze za jakýsi signál, že má smysl přistoupit ke skutečným testům. Věrohodnost matematického/fyzikálního popisu skutečnosti závisí totiž pouze na autorech práce a nelze implicitně předpokládat, že jsou modely dostatečně podobné fyzikální realitě, a že autoři zohlednili při návrhu všechny důležité aspekty. Ve studovaných publikacích probíhá vyhodnocení jak na simulačních datech [82, 31, 44, 29, 71, 51, 48], tak na reálném systému [32, 11, 19, 24, 13, 9, 46, 2, 41, 27]. U reálných experimentů je však v mnoha případech využíváno již hotového zakoupeného řešení včetně řídicí aplikace, většinou s využitím MATLABu [56, 13, 46, 45, 62, 21]. Takovéto sestavy mají vždy omezené možnosti při snaze implementovat vyřešené zadání do reálných podmínek, rozšířit model o další senzorický systém, nebo vyměnit některé jeho díly. Část uvedených reálně postavených modelů obsahuje vlastní návrh, většina ale využívá pro zjištění pozice kuličky primitivní metodu s odporovou dotykovou podložkou a mikropočítač, takže mají velmi úzký rozsah použití.

2.4 Sériový průmyslový robot vs Kulička na nakloněné rovině

Doposud odkazované publikace byly zpravidla navázány na systém „Ball & Plate“ - Kulička na nakloněné rovině. K této struktuře je publikováno velké množství ověřených strategií řízení pohybu, ale paralelní kinematická konfigurace, kterou obsahují bez výjimky všechny odkazované systémy, nemá s chováním sériového průmyslového robotu mnoho společného. Pro možnost využití dosavadních výsledků takovéto standardní topologie k testům použitelnosti na řízení robotu, byl uvažován systém, který se principem funkce podobá standardnímu modelu kuličky na nakloněné rovině, ale svými dynamickými projevy se blíží chování sériového průmyslového robotu. Výsledkem je právě v této práci od základu vytvořený model, jehož možnosti řízení, použití a dalšího rozšíření končí až na možnostech jeho uživatelů.

Mezi ověřené algoritmy regulace na průmyslovém robotu se sériovou kinematickou strukturou patří, mimo standardní kaskádní regulace složené z kombinace P(I)(D) regulátorů, např. sliding-mode (regulace v klouzavém režimu) [7, 61], computed-torque (výpočet momentů) [58, 37, 26, 12, 4, 74, 61], neuronové sítě [73, 74], speciální kombinace různých přístupů [26, 23] a mnoho dalších.

Testovat strategie řízení přímo na průmyslovém robotu je obecně problém, protože žádný současný výrobce nedovolí uživateli vstoupit ani do samotné regulační struktury robotu, natož zavézt vlastní metodu řízení. Publikace, zabývající se návrhem a vyhodnocením regulátorů přímo na průmyslovém robotu, obsahují z velké části pouze simulační ověření [73, 37, 4, 74, 23, 58, 61, 26], nebo testy na školních výukových modelech. Ty jsou buď zakoupeny jako hotová řešení [12, 35], nebo jsou navrženy a postaveny přímo autory práce [7, 3]. Z daleka však nedosahují typové dynamiky pohybu průmyslového robotu. Zařízení, které se dají považovat po všech stránkách za ekvivalent průmyslových robotů, a umožňující libovolný zásah uživatele do řídicího systému, jsou většinou výsledkem několikaletého vývoje celých týmů zabývajících se robotikou, konstrukcí, elektrotechnikou, programováním aj. Není tedy v možnostech autora této práce postavit podobnou robotickou strukturu, a proto bylo k řešení přistoupeno vlastní cestou.

3 CÍL PRÁCE

Hlavním cílem této dizertační práce je navrhnout soustavu se sériovou kinematickou strukturou a dynamickým chováním podobným průmyslovému robotu, na základě aktuálně dostupných a používaných technologií a principů, za účelem implementace a porovnání moderních algoritmů řízení pohybu. Celý systém musí být otevřený (jak zdrojový kód, tak konstrukční i elektrotechnické prvky), modulární a připravený na další rozšíření či inovaci dílčích komponent. Záměrem tedy není vytvořit pouze model simulující dynamické chování průmyslového robotu a porovnat navržené řídicí strategie, ale především reálně postavit systém, umožňující skutečné ověření různých přístupů souvisejících s řízením pohybu. Jsou jimi např. pokročilé algoritmy strojového vidění, využití neuronových sítí, multiprocesorové zpracování dat, aplikace specifických regulačních struktur, uplatnění výsledků matematického/fyzikálního modelování, vzdálené řízení, zpracování reálně naměřených dat a mnoho dalších.

Po návrhu typové struktury, sestrojení modelu v CAD, jeho výrobě, sestavení, osazení elektrotechnickými prvky, zprovoznění, volbě a naprogramování řídicí aplikace využívající strojové vidění, se bude tato práce zabývat porovnáním autonomní kaskádní P(I)(D) regulace (jakožto reprezentanta nejrozšířenější používané regulační struktury) s alespoň jednou vhodnou alternativou řízení pohybu. Na základě těchto výsledků bude poté zkoumáno, zda-li není možné dosáhnout v současné době výkonných výpočetních prostředků stejné, či lepší kvality/robustnosti regulace využitím moderních přístupů k řešené problematice, s cílem minimalizovat výkonové/materiálové/cenové požadavky. K dosažení hlavního cíle bude nutné splnit následující body zadání:

- Navrhnout typovou strukturu reálného modelu, na kterém bude probíhat testování různých strategií řízení - požadavek na nelinearitu a nestabilitu, kinematika i dynamika musí být podobná aktuálně používaným typům sériových průmyslových robotů.

- Sestrojit konstrukční a fyzikální model navržené robotické struktury; konstrukční 3D CAD model pro pevnostní studie a samotnou výrobu, fyzikální model prozatím pouze pro analýzu a simulaci dynamiky chování systému

za účelem následného výběru potřebných akčních členů.

- Vyrobit model s využitím aktuálních technologií - 3D tisk.
- Vybrat vhodné akční členy, servozesilovač, zdroje, řídicí počítač, kamerový systém a další nezbytné komponenty; vše sestavit a zprovoznit.
- Odvodit zjednodušený matematický model systému a upravit fyzikální model do podoby vhodné pro testování navržených řídicích algoritmů (detailní model včetně odvalování kuličky, tření, pružnosti, tlumení apod.).
- Provést identifikaci použitého servozesilovače, konkrétně zachované proudové smyčky, a navrhnout vhodnou aproximaci vzhledem k dynamice pohybu celého systému.
- Vyvinout řídicí aplikaci s grafickým rozhraním v jazyce C++.
- S použitím kamerového systému pro zjištění aktuální polohy řízeného objektu vybrat a implementovat vhodné algoritmy zpracování obrazu, zprovoznit komunikaci mezi servozesilovačem a řídicím počítačem s vhodně zvoleným komunikačním protokolem.
- Použít odvozený matematický model k návrhu různých metod řízení nelineárních systémů a ty otestovat nejprve na fyzikálním modelu, poté na reálném systému.
- Vzájemně porovnat navržené regulátory a zhodnotit jejich použití v reálném prostředí na skutečném systému.

4 ZVOLENÉ METODY ZPRACOVÁNÍ

V této kapitole budou postupně popsány jednotlivé kroky při sestavení kompletního mechatronického systému, od samotného návrhu typové struktury, až po konečné řízení celé sestavy. Jsou požity současné přístupy k řešení podobně komplexních problémů, jako je např. 3D CAD modelování, prototypování s využitím 3D tisku, uplatnění fyzikálního modelu nejen při simulacích, strojové vidění s využitím kamerového systému, distribuované řízení, multiprocesorové zpracování dat, aplikace kinematických transformací pro rekonstrukci polohy v reálném čase a mnoho dalších.

Samotný návrh a kompletace hardwarové části modelu je nedílnou součástí práce, a proto bude této pasáži vzhledem k nezanedbatelné časové režii rovněž věnován odpovídající rozsah.

4.1 Návrh typové struktury

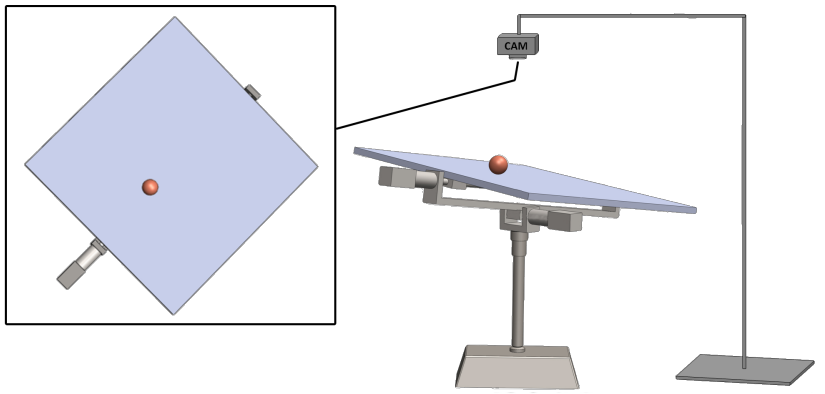
V současnosti se používá pro konstrukční návrh výhradně 3D CAD software. Mezi asi nejznámější patří AutoCAD, Invertor, CATIA, SolidWorks atd. Zde je použitý poslední zmíněný - SolidWorks.

Tento software má přívětivé uživatelské rozhraní, srozumitelnou dokumentaci, možnosti animace, simulace a statické/dynamické analýzy. Pomocí dostupného plug-inu je možné jej propojit s MATLABem a využít tak speciální toolbox pro mechanické simulace.

Využití 3D CAD modelu má nespočet výhod nejen při konstrukční fázi, ale také při samotném návrhu zákona řízení. CAD model obsahuje kromě přesných geometrických rozměrů také materiálové vlastnosti, a v případě sestav i vazby mezi jednotlivými díly. To umožňuje poměrně jednoduché získání důležitých parametrů pro návrh zákona řízení, jako jsou např. hmotnosti, matice setrvačnosti podsestav, vektory polohy těžišť jednotlivých dílů v libovolné souřadné soustavě, kinematické limity kloubů a další. Model je možné také vyexportovat do formátů, které mají další neopomenutelné využití. Patří mezi ně např. STL (3D tisk), SimMechanics (fyzikální model v MATLABu), VRML/WRL (grafický formát pro aplikace virtuální reality - např. pro 3D Animation Toolbox v MATLABu) atd.

[75, 36].

CAD model je využíván v podstatě od počáteční fáze konstrukčního návrhu, až po konečnou fázi samotného řízení celého systému. Z tohoto důvodu by měl být nedílnou součástí každého komplexního projektu, i v případech, kdy není nutné systém reálně stavit.



Obr. 4.1 CAD model systému - typová struktura, prvotní fáze návrhu

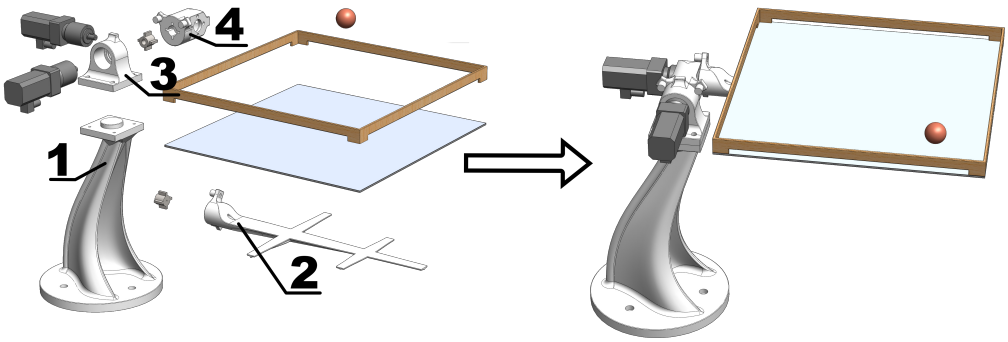
Nejprve byla tedy navržena typová struktura pro ověření základního chování modelu a docílení požadovaných parametrů - systém dosahuje dostatečných rychlostí, aby se projevily nelinearity a vzájemné interakce mezi aktuátory; kinematika i dynamika je podobná aktuálně používaným typům průmyslových robotů - Obr. 4.1.

Po navržení typové struktury bylo nutné ověřit, zda-li chování modelu dosahuje očekávaných předpokladů. To se realizovalo v MATLABu po exportu CAD modelu do SimMechanics toolboxu (viz kapitola 4.8.2), pomocí inverzní dynamické úlohy, která je zde implementovaná.

4.2 Konstrukční návrh

Dalším krokem je překreslení sestavy do konstrukční podoby, aby bylo možné vše vyrobit. Zde přišly na řadu výrazné úpravy, především v uložení a napojení akčních členů. Již při návrhu byla kladena zřetel na možnost výroby celého modelu pomocí 3D tiskárny, proto bylo nutné dodržet odpovídající podmínky.

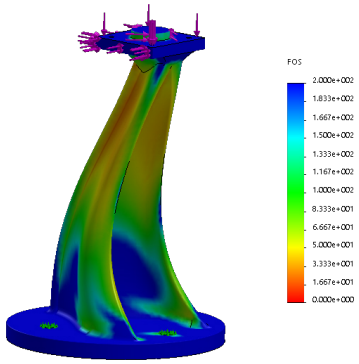
Výsledkem je model se sériovou kinematickou strukturou a částečně uvolněným řízeným objektem s 5 stupni volnosti (DOF) - Obr. 4.2.



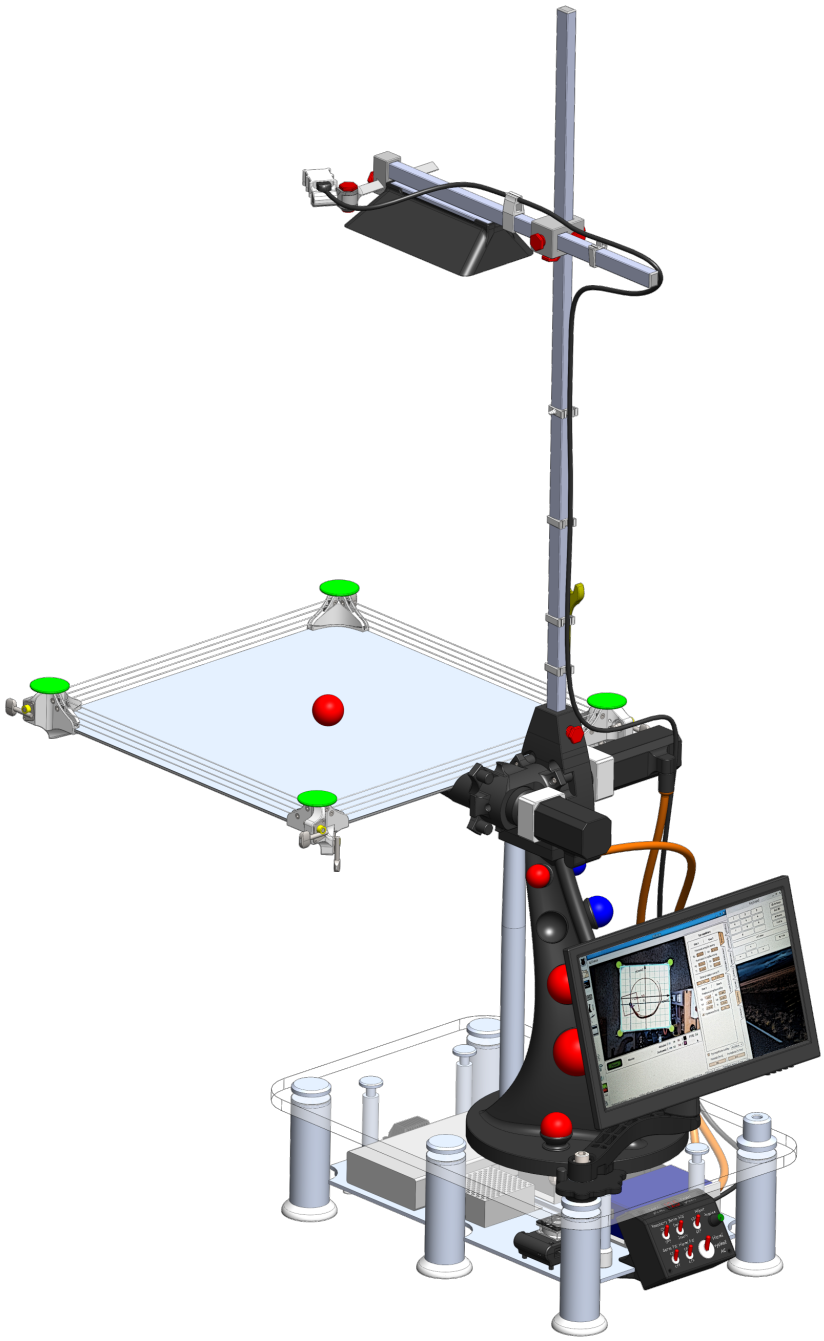
Obr. 4.2 CAD model systému - konstrukční návrh základních částí sestavy

Soustava bude řízena prostřednictvím vázané rotační dvojice zajišťující rozklad vektoru gravitační síly tak, aby jeho gradient zajistil žádaný pohyb řízeného objektu. Systém je známý pod názvem „Ball & Plate“ - Kulička na nakloněné rovině. Jeho konstrukční řešení je však zcela odlišné od běžně prezentovaných modelů řešící úlohu regulace kuličky - [22, 6, 27, 64, 2, 8, 33, 76, 81, 68, 9, 40, 59, 62, 24, 56, 77, 45, 32, 19, 21, 14, 13, 51, 48, 71, 10, 52, 70, 34, 46, 11, 29, 44, 31, 82, 47].

Tyto systémy mají ve všech případech paralelní kinematickou strukturu, a tedy rozdílnou dynamiku pohybu nakloněné roviny. Z valné části jsou hmoty pohyblivých částí vůči hmotě kuličky zanedbatelné a snímání její polohy je často řešeno pomocí odporové dotykové podložky.



Obr. 4.3 Statická simulace namáhání základny při maximálních momentech



Obr. 4.4 CAD model systému - konečný konstrukční návrh

Statickými pevnostními simulacemi bylo pro každý díl alespoň orientačně zjištěno, jaké části je nutné vyztužit a v jakém místě bude docházet k největšímu namáhání. Získání přesných hodnot dovolených zatížení je z důvodu použitého 3D tisku při výrobě v podstatě vyloučené, protože vrstvením materiálu není díl dokonale homogenní a záleží na mnoha faktorech ovlivňujících výslednou pevnost/pružnost. Jsou jimi např. teplota tiskové hlavy, natočení dílu při tisku, vzor výplně, výška vrstvy, použitý průměr trysky, okolní podmínky, stáří, stav a výrobce tiskové struny, tiskový materiál, objemový průtok materiálu, přesahy vrstev při výplni, vibrace tiskové plochy a mnoho dalších - [38, 42, 69, 17, 65, 54].

Konečná verze CAD modelu je zobrazena na Obr. 4.4. Obsahuje všechny významné komponenty utvářející celý systému. Ve většině případů byly prvky vymodelovány kvůli jejich výrobě (zpravidla 3D tisku), ale také např. z důvodu vizuální představy o jejich rozmístění a následného upevnění, dostupnosti ovládacích prvků, omezení/dorazů pohyblivých částí apod.

4.3 Elektrotechnické vybavení

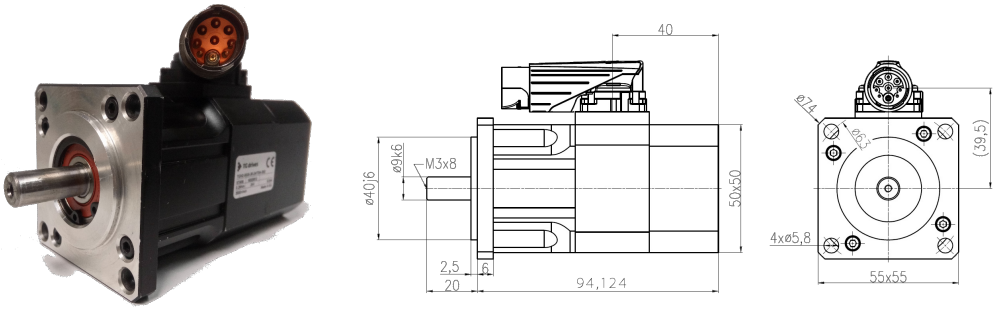
V této kapitole budou popsány použité elektrotechnické prvky, jejich parametry a důvod výběru. Hardware byl volen vždy s ohledem na možnosti budoucího rozšíření. V neposlední řadě byla brána v potaz samozřejmě i cena jednotlivých komponent, respektive poměr cena/užitná hodnota.

4.3.1 Elektromotor

Hlavní částí celého pohybového systému jsou akční členy, v tomto případě elektromechanické - přeměňující elektrickou energii na mechanickou práci. Nejpoužívanější servopohony, tedy elektromotory s řídicím systémem, jsou varianty trojfázových synchronních motorů s permanentním magnetem v rotoru (PMSM). Tyto servomotory se používají pro precizní polohové řízení, a spolu s řídicí jednotkou (servozesilovačem) jsou ideální volbou pro zařízení, kde není nutné řešit samotný návrh výkonové elektroniky.

U těchto motorů je v současnosti používané vektorové řízení - převod z 3-fázového sinusového systému do 2-souřadnicového časově nezávislého souřadného systému d, q pomocí Clarkovy a Parkovy transformace [49, 43, 78]. Složka d se

podílí na vzniku magnetického toku, složka q zase na vzniku točivého momentu motoru. Regulací proudu I_q je tedy přes momentovou konstantu přímo řízen moment ve vzduchové mezeře stroje, potažmo na hřídeli motoru (se zanedbáním tření v ložiscích a zátěžného mementu od momentu setrvačnosti rotoru).



Obr. 4.5 Použité elektromotory od brněnské firmy TG Drives [66]

Jako akční členy byly vybrány synchronní motory s permanentním magnetem v rotoru od brněnské firmy *TG Drives* - Obr. 4.5. Tato společnost navíjí motory i dle požadavků zákazníka a za nabízenou cenu/kvalitu by se jen těžko hledal konkurent, nejen na českém trhu. Hlavním důvodem tohoto konkrétního výběru byl ale fakt, že nabízí i vlastní servozesilovač, u kterého je možné odpojit nadřazené regulační smyčky (polohovou a rychlostní) a ponechat pouze smyčku proudovou. Tuto smyčku je možné dále parametrizovat, struktura proudového regulátoru však musí zůstat zachována - použité vektorové řízení není ovšem důvod modifikovat.

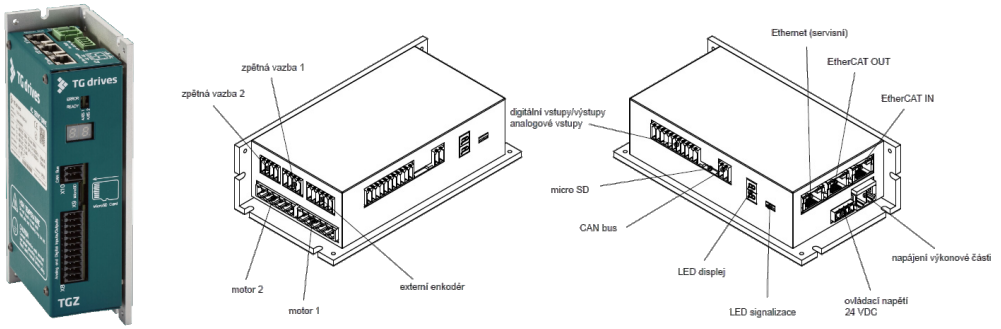
Pro 1. DOF byl vybrán motor s klidovým točivým momentem 0,75 Nm, pro 2. DOF byla zvolena méně výkonná varianta s momentem 0,28 Nm. Oba motory jsou z řady TGN2 v jedno-konektorovém provedení na bezpečné napětí. Mají 6 pólový rotor a neobsahují elektromagnetickou brzdu. Jako snímač polohy natočení je použitý více-otáčkový 17 bitový enkodér, jehož hodnota je do servozesilovače přenášená pomocí rychlé digitální komunikace. Datové žíly jsou tedy pouze dvě a jsou integrované do silového kabelu. Detailní informace o akčních členech jsou uvedeny v Tab. 4.1.

Tab. 4.1 Parametry použitých elektromotorů řady TGN2 [66]

Vstupní napětí			**24 VDC**	**36 VDC**
Magnety Nd-Fe-B			**TGN2-0028**	**TGN2-0075**
Klidový moment	M_0	Nm	0,28	0,75
Klidový proud	I_0	A	8.04	7,7
Jmenovitý moment	M_N	Nm	0,25	0,72
Jmenovité otáčky	n_N	$min.^{-1}$	3000	2500
Jmenovitý výkon	P_N	W	104	188
Jmenovitý proud	I_N	A	7,8	7,8
Maximální moment	M_{max}	Nm	1,1	2,7
Maximální proud	I_{max}	A	36	31
Max. otáčky mech.	n_{max}	$min.^{-1}$	12000	12000
Momentová konst.	K_m	Nm/A	0,04	0,1
Napěťová konst.	K_E	$\frac{V}{1000min.}$	2,2	5,9
Počet pólů motoru	$2p$	-	6	6
Odpor dvě fáze	R_{2Ph}	Ω	0,43	0,66
Indukč. dvě fáze	L_{2Ph}	mH	0,42	0,89
Vlastní mom. set.	J	$kgcm^2$	0,05	0,09
Hmotnost	m	kg	0,76	1,1

4.3.2 Servozesilovač

Výkonovou elektroniku (servozesilovač) navrhují výrobci přímo na konkrétní řadu motorů, zahrnuje různé proudové a přepěťové ochrany, komunikační rozhraní, uživatelské vstupy a výstupy, ověřený systém chlazení, brání demagnetizaci rotoru, apod. Také obsahuje rychlou proudovou smyčku, jejíž řízení z nadřazeného systému by mohl být problém - z pohledu dodržení pevné periody vzorkování (běžně okolo 50 μs). Nastavení proudového regulátoru navíc nezáleží na připojené zátěži, může být tedy nastaven jednou a poté stačí podle zátěže přenastavovat jen nadřazené smyčky. Z výše uvedených důvodů je proto logické využít tento ideálně navržený servozesilovač a nezaobírat se návrhem výkonové části, která lze jen těžko optimalizovat, pokud to není přímo cílem práce.
Konkrétně byl vybrán servozesilovač řady TGZ-48 od stejné firmy, jako elektromotory (dále již motory) - Obr. 4.6. Je navržen pro řízení dvou os nízkonapěťových verzí do 48 VDC, nepřesahujících nominální proud 13 A pro jednu osu.



Obr. 4.6 Vybraný servozesilovač/řídicí jednotka od firmy TG Drives [66]

Aktuátory mohou patřit do rozdílných typových řad s odlišným točivým momentem, všechny důležité parametry si zákazník nastaví v dodávané uživatelské aplikaci. Podrobnější informace o servozesilovači jsou uvedeny v Tab. 4.2.

Tab. 4.2 Technická data použitého servozesilovače řady TGZ-48 [66]

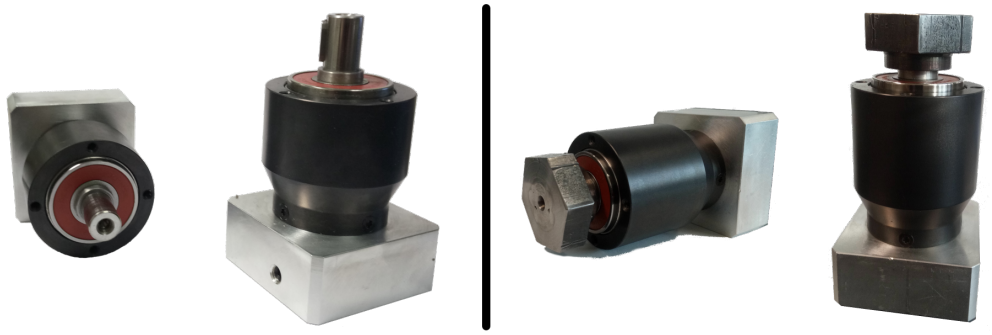
Parametry	**TGZ-D-48-13/26**
Ovládací napětí	24 V DC ±10 %
Výkonové napájecí napětí	6 - 48 V DC
Instalovaný příkon pro provoz S1	2 x 500 W
Trvalý proud na jednu osu	13 A
Trvalý celk. proud při provozu 2 os	26 A
Maximální výst. proud (max. cca 5 s)	2 x 26 A
Ztráty při jmenovité zátěži	20 W
Komunikace	
CAN	4pin S2C-SMT 3.50
ETHERCAT IN/OUT	100/1000 Mb/s, 2 x RJ45
ETHERNET UDP (servis)	100/1000 Mb/s, RJ45
Vstupy/Výstupy	
2 AI, 8 DI, 6 DO - využití jazyka C	1 x 22pin S2C-SMT 3.50
Signalizace	
LED displej	2 x 7 segment LED
LED signalizace (osa 1 a 2 zvlášť)	zelená, červená
Prostředí	
Teplota okolí	0 - 40 °C
Relativní vlhkost	max. 85 % (bez kondenzace)

Uživatelská aplikace *TGZ GUI* nabízí nastavení připojení, monitoring snímaných veličin jednotlivých os v reálném čase, bohatou parametrizaci připojených motorů, konstant regulátorů, náběhových/doběhových ramp, módů atd., vzdálené řízení polohy i rychlosti v různých režimech, grafické zobrazení či záznam dat volitelných veličin v bloku nazvaném jako osciloskop (možnost nastavení periody vzorkování, triggeringu, odečítání hodnot, ...) aj. Z pohledu této práce je ovšem hlavním nastavitelným parametrem mód servozesilovače, ovlivňující režim, ve kterém probíhá řízení - regulace polohy, rychlosti, nebo proudu. Výrobce v konečném důsledku neumožňuje zákazníkovi měnit pouze konstanty regulátoru, ale dokonce i jeho strukturu (zařazením vlastní nadřazené regulační struktury nad proudovou smyčku).

4.3.3 Převodovky

I když je možné nechat si motor speciálně navinout na nižší jmenovité otáčky, stále je pro tyto účely zbytečně rychlý a při dodržení kompaktní velikosti má malý točivý moment. Proto je v tomto případě vhodné využít mechanické převodovky „do pomala“. Převodový poměr udává násobnost zvětšení výstupního točivý momentu, resp. zmenšení výstupní úhlové rychlosti.

Na trhu je velké množství různých typů převodovek s odlišným principem funkce, převodovým poměrem, rozměry a především cenou. Při zachování velikosti, hmotnosti, tuhosti a únosné vůle mezi ozubením, je planetová převodovka v poměru cena/využití nejpřívětivější. Její cena je při podobném převodovém poměru např. oproti cykloidní převodovce zhruba na $^1/_{10}$.
Od stejné firmy jako servomotory jsou pořízeny také převodovky, konkrétně planetové s převodovým poměrem 1:25 do pomala - Obr. 4.7. Jsou to ekonomické verze planetových převodovek s 2 stupni a vůlemi max. 19 arcmin, přechod mezi hladkou výstupní hřídelí motoru a svěrnou vstupní přírubou převodovky je bezvůlový. Podrobnější informace jsou uvedeny v Tab. 4.3. Uvedená rotační vůle se při použití na tomto modelu v nejvzdálenějším bodě od osy rotace (Obr. 4.4 - roh nakloněné roviny) projevuje jako translační vůle o velikosti cca 4,5 mm. Tato hodnota udává nejhorší možný scénář při přechodu obou akčních členů do reverzního pohybu, pokud v celém systému nebude žádná jiná vůle. Reálně k



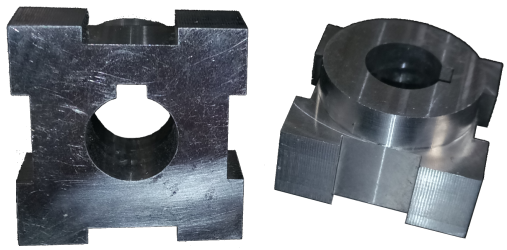
Obr. 4.7 Instalované planetové převodovky s neosazenou (1:10 vlevo) a osazenou (1:25 vpravo) hřídelí

Tab. 4.3 Parametry použitých planetových převodovek řady SG 050 [66]

Parametry		**SG 050**
Počet stupňů		2
Převod		25
Průměr vstupní hřídele		max. 11 mm
Jmenovité/maximální vstupní otáčky	$min.^{-1}$	5 000 / 10 000
Jmenovitý výstupní moment	Nm	13
Maximální výstupní moment	Nm	21
Nouzový výstupní moment	Nm	36
Torzní tuhost	$Nm/arcmin$	0,8 - 1,1
Moment setrvačnosti	$kgcm^2$	0,014 - 0,026
Maximální vůle	$arcmin$	< 19
Účinnost	%	95
Provozní teplota	$°C$	- 25 ... + 90
Životnost	hod	30 000
Hmotnost	kg	0,9
Hlučnost	$db(A)$	58
Radiální zatížení	N	700
Axiální zatížení	N	800

tomuto případu ale nedochází, protože se systém v první ose, kde je projev vůle největší, nepohybuje nikdy okolo ustáleného stavu a gravitační účinky tak zajistí markantní redukci stanovené vůle.

Výstupní hřídel převodovky je typu pero-drážka, a tak byly u prototypu vy-



Obr. 4.8 Spojky přechodu hřídele

robeny protikusy na tento typ hřídele - Obr. 4.8. Základní tvar byl vyrobený pomocí CNC stroje, přesná drážka zase s využitím elektrojiskrového obrábění (tzv. „drátořez“). Tento typ spoje se ale i přes použití tvrdého materiálu dural 7075 na první ose vymačkal, a tím se několikanásobně zvětšila vůle. Proto musel být použitý jiný typ spoje, i s ohledem na jeho opakovatelnou rozebiratelnost bez použití speciálního nářadí, což u tohoto spoje splněno nebylo. Napojení výstupní hřídele převodovky s perem přímo na plastový díl je vyloučeno, musí zde být mezikus zajišťující velkou styčnou plochu mezi plastovým dílem, aby nedocházelo k vymačkávání materiálu a zároveň musí být mezikus napojený na výstupní hřídel převodovky pokud možno bezvůlově.

Ideálním řešením by byl svěrný spoj, který je použitý na spojení přechodu motor-převodovka a funguje na principu suchého tření. Na výstupu převodovky je ovšem 25x větší točivý moment a pro zamezení prokluzu musela být vyrobena masivní svěrná spojka se speciálním uchycením v plastovém dílu. Při testech ale ve spoji docházelo vlivem momentových rázů k mikro-prokluzu, na základě kterého se kumulovala úhlová chyba. Jelikož není cílem této práce řešit konstrukční detaily, bylo k problému přistoupeno poněkud šalamounsky, viz Obr. 4.7 - vpravo. Z výstupní hřídele převodovky je vyjmuto pero a vyrobená ocelová spojka ve tvaru pravidelného pětiúhelníku s kruhovým otvorem ve středu je na hřídel nalisována s využitím tepelné roztažnosti materiálu - na zmrazenou hřídel převodovky je nasunuta rozžhavená spojka a po vyrovnání teplotního rozdílu vznikne těžko rozebiratelný spoj. Stále možný prokluz byl dále pojištěn svarem. Spojka má dostatečnou styčnou plochou mezi plastovým dílem a vhodný tvar k 3D tisku odpovídajícího otvoru v protikusu. Spoj mezi plastovým dílem je rozebiratelný a nebyla doposud pozorována žádná vůle. Nevýhodou tohoto řešení

je pochopitelně nerozebiratelný spoj mezi hřídelí převodovky a spojkou, což ale nikterak nebrání plnohodnotnému použití.

4.3.4 Napájení

Na základě zvolených motorů s bezpečným napětím bylo nutné pořídit spínaný zdroj 230VAC/24-48VDC pro splnění požadovaného napájecího napěťového rozsahu. Ze simulací na fyzikálním modelu (kapitola 4.8.2) dále vyplývá, že zdroj musí být schopen dodat ve špičce 20 A. Pro výkonovou a řídicí část servozesilovače není doporučováno používat společné napájení a má navíc jiný napěťový rozsah, než nadřazený řídicí počítač.

Obr. 4.9 Použité napájecí zdroje a proudová ochrana-jištění

V řízeném systému jsou tedy použity celkem tři spínané napájecí zdroje. Jeden zdroj 24V/20A napájí výkonovou část servozesilovače, druhý zdroj 24V/1,5A je pro napájení řídicí části servozesilovače a třetí zdroj 5V/7A pro řídicí počítač s kamerou. Z důvodu otevřené desky s elektronikou a související bezpečnosti byl dále instalován i proudový chránič s nadproudovou ochranou (vypínací charakteristika typu B se jmenovitým proudem 10 A a vypínacím reziduálním proudem 30 mA) - Obr. 4.9. U zdroje 24V/20A je využito implementované vzdálené ON-OFF ovládání a z důvodu vysoké hlučnosti aktivního chlazení byly vyměněny stávající ventilátory za kvalitní tiché (Noctua NF-A4x20 FLX). Dnešní spínané zdroje mají integrované ochrany proti přetížení, zkratu, přepětí, přehřátí a mají korekci výstupního napětí. Není proto potřeba se o žádnou z výše uvedených

eventualit starat.

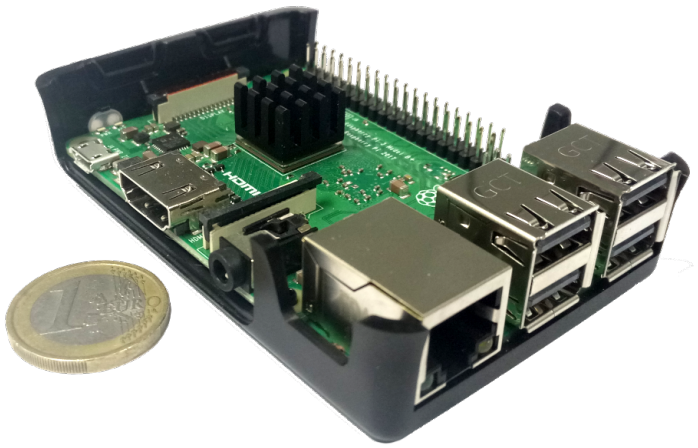
Tab. 4.4 Vybrané parametry použitých napájecích zdrojů a proud. ochrany

Parametry	**MEAN WELL SP-480-24**
Vstupní napětí	85 ∼ 264 VAC (47 ∼ 63 Hz); 120 ∼ 370 VDC
Výstupní napětí	24 V
Nominální proud	20 A
Vzdálené ovládání	RC+/RC-: 0 ∼ 0,8V=power on ; 4 ∼ 10V=power off
Ochrany	proudová, přepěťová, teplotní
Hmotnost	1,7 kg
Parametry	**TDK-Lambda LS-35-5**
Vstupní napětí	85 ∼ 264 VAC (47 ∼ 63 Hz); 120 ∼ 370 VDC
Výstupní napětí	5 V
Nominální proud	7 A
Ochrany	proudová, přepěťová
Hmotnost	0,27 kg
Parametry	**MEAN WELL RS-35-24**
Vstupní napětí	85 ∼ 264 VAC (47 ∼ 63 Hz); 120 ∼ 370 VDC
Výstupní napětí	24 V
Nominální proud	1,5 A
Ochrany	proudová, přepěťová
Hmotnost	0,3 kg
Parametry	**OLE-10B-1N-030AC**
Provozní napětí	230 VAC
Vypínání	charakteristika typu B
Jmenovitý proud	10 A
Reziduální proud	30 mA
Počet pólů	1 + N-pól
Hmotnost	0,285 kg

4.3.5 Řídicí počítač

Celková dynamika řízení je pouze tak rychlá, jak rychlá je proudová smyčka. Pokud řízení této smyčky obstarává servozesilovač na to navržený, řídicímu počítači odpadá nutnost držet pevnou vzorkovací periodu na hodnotách nízko pod

hranicí 1 ms (reálně 25 - 50 µs), což je v případě použití operačního systému bez reálného času problém. Zvládat takto vysokou vzorkovací frekvenci je obecně složité i pro komunikační rozhraní s protokolem využívající ty nejnižší vrstvy referenčního modelu ISO/OSI. Požadované vzorkovací periody zbývajících nadřazených smyček budou dle prvních odhadů nad hodnotami 2 - 5 ms (reálně při kaskádní regulační struktuře postačuje 10 - 20 ms), což pro výpočet regulátorů a následný přenos přes komunikační rozhraní zvládne v současnosti vykonat i běžný kompaktní počítač se standardním operačním systémem.



Obr. 4.10 Řídicí počítač Raspberry Pi 3B+

Jako řídicí počítač byl tedy zvolen Raspberry Pi 3B+, který bez problémů zvládá vykonávat požadované výpočty - Obr. 4.10. Na tomto 4-jádrovém kompaktním počítači je Raspbian - operační systém odvozený z Debianu (Linux) pro Raspberry Pi [67]. Výhodou je možnost použití knihovny pro zpracování obrazu (OpenCV), snadné připojení kamery a práce s ní, programování v přehledném grafickém vývojovém prostředí a knihovní podpora např. pro komunikační protokoly, tvorbu grafických aplikací, vícevláknové zpracování programu atd. Raspberry bylo osazeno pasivními chladiči na všechny 3 základní čipy (procesor, IO řadič, RAM) a namontováno do speciálně navrženého a vyrobeného pouzdra s aktivním chlazením (Noctua NF-A4x10 FLX).

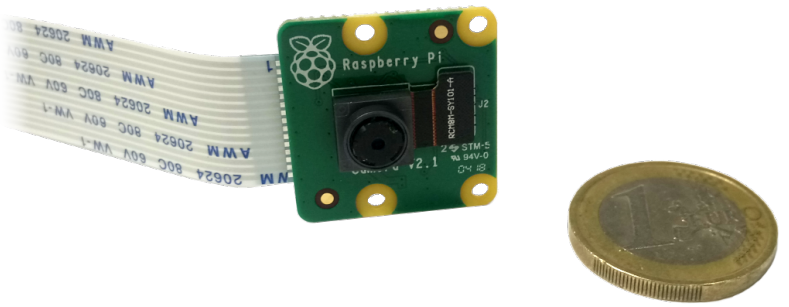
Tab. 4.5 Parametry kompaktního počítače Raspberry Pi 3B+

Parametry	Specifikace
Procesor	Broadcom BCM2837B0, Cortex-A53 64-bit, 1,4GHz
Operační paměť	1GB LPDDR2 SDRAM
Základní výbava	Bluetooth, micro SD, micro USB, Wi-Fi
Další	4 x USB 2.0, HDMI, CSI, DSI, RJ-45, jack 3,5 mm
Napájení	5 V / 2,5 A

4.3.6 Snímání skutečné polohy řízeného objektu - kamera

V řešeném případě je potřeba nezávisle snímat skutečnou polohu řízeného objektu, protože má s řízenou sestavou pouze jednu neholonomní kinematickou vazbu omezující jeden stupeň volnosti - translaci ve směru kolmém na podložku - tečná vazba kuličky s podložkou. Z tohoto faktu vyplývají v podstatě 3 možnosti, jak kuličku snímat: kontaktně, opticky, nebo její polohu rekonstruovat. Kontaktně je myšleno dotykem kuličky o podložku, na které bude umístěna detekční vrstva (ať už kapacitní, či odporová). Tato metoda má určitá úskalí. Obecně je problém získat dotykovou vrstvu o žádaných rozměrech, výstup bývá značně zašuměný a je potřebná dodatečná elektronika pro výčet a zpracování výstupních signálů. Dále musí mít kulička v případě kapacitní vrstvy elektricky vodivý povrch, v případě resistivní vrstvy musí mít dostatečnou hmotnost, aby byla schopna prohnout pružnou membránu. Optické snímání standardně vykonává CCD/CMOS kamera. Je to moderní metoda strojového vidění, která je stále více používána a má širokou podporu ze strany open-source knihoven. Jediným významným omezením jsou světelné podmínky, které jdou ovšem v případě nutnosti přizpůsobit. Rekonstrukce polohy přes lineárního/nelineárního pozorovatele je další možností snímání. Na základě snímaného proudu a přesné znalosti dynamického modelu je možné detekovat polohu řízeného objektu, pokud má změna jeho pozice významný vliv na dynamické chování celé sestavy. V mnoha případech ovšem nedosahuje kvůli výše zmíněným limitacím dostatečné přesnosti, proto je její použití vždy nutné ověřit.

Jako nejvhodnější způsob zjištění skutečné pozice řízeného objektu bylo zvoleno optické snímání - Raspberry Pi v2 kamerový modul, komunikující s řídicím



Obr. 4.11 CMOS 8 Mpx Raspberry Pi v2 kamera

počítačem po CSI. Tato kamera je typu CMOS s rozlišením senzoru 8 Mpx a velmi nízkou distorzí čoček. Není proto potřeba zbavovat se v předzpracovaní obrazu často se vyskytujícího „soudkovitého“ zkreslení pomocí kalibračních vzorů se známými geometrickými vlastnostmi. Kameru vyvíjí stejná společnost jako řídicí počítač, proto je přizpůsobená pro práci s ním. Z řídicího programu je možné volit mezi různými módy, které ovlivňují výsledné rozlišení videa a tedy i snímkovací frekvenci. Dále je k dispozici základní parametrizace, jako je nastavení jasu, kontrastu, saturace, expozice, zisku, apod. Vypnutí veškerých automatických funkcí, jako automatické nastavení expozice, zisku, balance bílé, či automatická volba video-módu je ve zpracování obrazu v reálném čase v podstatě nutností a tato kamera zakázaní všech uvedených funkcí podporuje. Zaostření obrazu je zde manuální, jinak by také bylo nutné jej potlačit.

Tab. 4.6 Vybrané parametry použité Raspberry Pi v2 kamery

Parametry	**Specifikace**
Senzor	Sony IMX219, CMOS, 3280 × 2464 pixelů
Velikost senzoru	3,68 x 2,76 mm
Ohnisková vzdálenost	3,04 mm (reálně 2,94 mm)
Horizontální zorné pole	62,2°
Vertikální zorné pole	48,8°
Video módy	1080p30fps, 720p60fps, 480p90fps
Video formát	raw h.264

Z důvodu dlouhého flex kabelu a možného rušení/snížení přenosové rychlosti

byla kamera doplněna oboustrannou redukcí na HDMI kabel. Pro zajištění alespoň základních světelných podmínek byl ke kameře navržen regulovatelný LED přísvit ovládaný z řídicího panelu o výkonu cca 15 W. Jako chlazení těchto LED panelů bylo využito hliníkového „lešení“ kamery - Obr. 4.4.

4.3.7 Zobrazovací jednotka

Pro zobrazení výsledků zpracování obrazu, vzdálené ovládání modelu, grafické výstupy, parametrizaci kamery, regulátorů a algoritmů zpracování obrazu, nastavení spojení mezi řídicím počítačem a servozesilovačem apod. je použitý dotykový displej ASUS VT168H velikosti 15,6".

Obr. 4.12 Použitý dotykový displej ASUS VT168H

Tab. 4.7 Parametry monitoru ASUS VT168H

Parametry	**Specifikace**
Úhlopříčka	15,6"
Rozlišení	1366 × 768 px
Typ panelu	TN
Obnovovací frekvence	60 Hz
Odezva	10 ms°
Jas	200 cd/m2
Kontrast	50M:1
Pozorovací úhly - vert./horizont.	65°/90°
Připojení	HDMI, VGA, USB

K řídicímu počítači je monitor připojen přes HDMI rozhraní, dotyková vrstva

supluje myš a je připojena pomocí USB. Displej má rozlišení 1366 x 768 bodů a je uchycen na navrženém a vyrobeném prototypu otočného magnetického držáku. Doplňující parametry jsou uvedeny v Tab. 4.7.

4.3.8 Ostatní

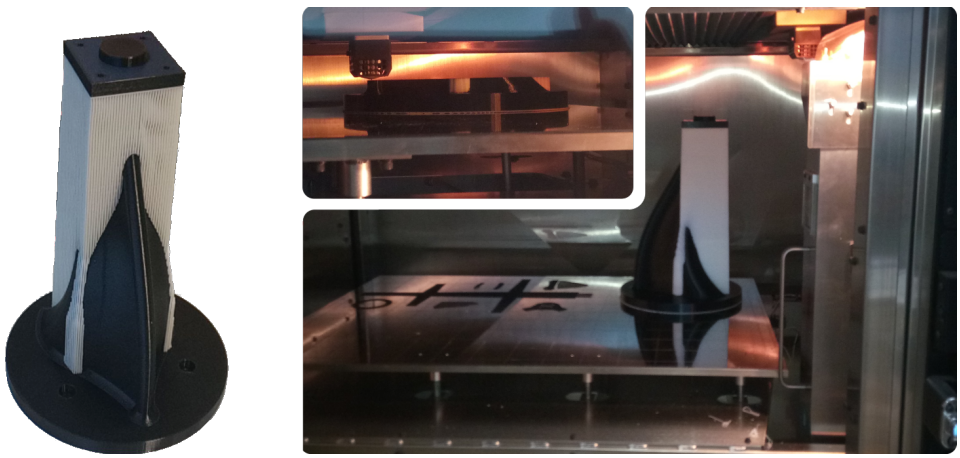
Součástí elektrotechnického vybavení modelu jsou také další podružné periferie, jako je napěťový regulátor pro regulaci LED přísvitu, gyroskopická myš s klávesnicí pro ruční řízení modelu a jako doplňkový vstupní prvek, hardwarový ovládací panel modelu s přepínáním řídicího počítače (Raspberry/externí), vypínači pro silovou i ovládací část aj., USB a HDMI rozbočovač, přídavný mini reproduktor a mnoho dalšího.

Obr. 4.13 Další periferie elektrotechnického vybavení

4.4 Prototyp

Aktuálně nejpožívanějšími a relativně levnými metodami prototypování jsou různé varianty 3D tisku. Tato technologie je dostupná i v segmentu pro domácí (hobby) použití s menší tiskovou plochou (3D modely do velikosti cca 200x200x200 mm) a je na výběr z desítek druhů tiskových materiálů a barevných variant. 3D tisk dosahuje přesnosti 50 µm a slití vrstev je při správném nastavení tiskárny velice dobré. Navíc je zde možnost částečné výplně dílů, což minimalizuje spotřebu materiálu a také hmotnost výsledného dílu. S podporami z rozpustitelného materiálu lze tisknout v podstatě libovolný 3D objekt, dokonce i pohyblivé nerozebiratelné sestavy, objekty v objektu atd.

Výhodou 3D tisku při prototypování je výroba navrženého, příp. přepracovaného dílu v podstatě na počkání. 3D CAD model je k dispozici z konstrukčního návrhu (kapitola 4.2), ten už je nakreslený s ohledem na požadavky 3D tisku (vůle, přesahy, přechody atd.), vyexportuje se do STL formátu, odešle na 3D tiskárnu a za několik minut/hodin je k dispozici nový díl na další testování.



Obr. 4.14 Velkoplošný 3D tisk základny modelu

Na velkoplošné 3D tiskárně byly vyrobeny základní 4 části, z nichž se dochovaly pouze dvě z nich (část 1 a 2 - Obr. 4.2). Zbylé dva díly při prvotním testování praskly a byly vytištěny znovu na hobby 3D tiskárně *Original Prusa I3 MK3S* - [30] z poddajnějšího materiálu a s dalšími vylepšeními (zdvojené mechanické

dorazy s jiným uchycením tlumičů, držák na lešení kamerového systému apod.). Na této tiskárně byly dále vytištěny všechny ostatní části modelu, jako jsou dorazy, tlumiče, držáky, kryty, krabičky, ovládací panel, napínáky, kalibrační vzory, těsnění, podložky a další.

4.4.1 První verze

První verze modelu sloužila ke zprovoznění základních pohybů nakloněné roviny bez řízení polohy kuličky. Nebyl zde žádný nadřazený řídicí systém, pouze servozesilovač, chybí uživatelské rozhraní, lešení s kamerovým systémem a přísvitem, proudové jištění/ochrana, pasivní chlazení, ovládací panel a mnoho dalšího.

Obr. 4.15 První reálná verze modelu *Kulička na nakloněné rovině*

Tento model sloužil pro výběr vhodných pohonů a převodovek, otestování zvolené volby uložení hlavních částí pohybových mechanismů, tuhosti a pevnosti navržené konstrukce, modulárnosti a možnosti dalšího rozšíření, ověření dostatečného výkonového nadimenzování aj.
Jako hlavní řídicí systém celé sestavy slouží servozesilovač a je využito regulační smyčky v něm integrované. Zde není možná kontrola nad průběhem regulace, na-

Obr. 4.16 Testovací zapojení první verze prototypu

stavování žádaných hodnot se provádí přes software dodávaný výrobcem, stejně tak jako parametrizace servozesilovače, grafické zobrazení průběhů zvolených veličin s periodou vzorkování až 50 µs atd. Podpůrný software běží na osobním počítači s operačním systémem Windows a je propojený se servozesilovačem přes Ethernet. Veškerá parametrizace a nastavení pro přípravu regulace systému pomocí zvoleného řídicího počítače probíhá s využitím tohoto softwaru.

Právě při testování maximálního požadovaného zrychlení praskly nosné části fixující oba aktuátory s převodovkami, jak již bylo zmíněno v předchozí kapitole (část 3 a 4 - Obr. 4.2). Také došlo k vymačkání styčných ploch spojek zajišťujících přechod mezi výstupem převodovky a plastovou přírubou. Hlavní nosná část (základna - část 1 - Obr. 4.2) při pohybu výrazně pružila, proto byla vyztužena hliníkovými tyčemi. Podstava/stůl s elektronikou byl kompletně přepracován, namísto dřevěného podstavce byl vyroben bytelný stůl z plexiskla, hliníkových tyčí a s chladící hliníkovou deskou určenou k montáži elektroniky a sloužící zároveň také jako vodivé pospojování a pasivní chlazení. Stůl byl dále vybaven základním ovládacím panelem a k tělu modelu bylo přimontováno lešení s kamerou a přísvitem. Systém je navržen jako modulární, většina součástí je lehce demontovatelná a celkově je model pojat jako vývojový a rozšiřitelný systém pro výukové, testovací a vědecké účely.

4.4.2 Současná verze

Poslední verze modelu obsahuje všechny výše uvedené komponenty a vylepšení, je však stále ve vývoji.

Obr. 4.17 Současná reálná verze systému *Kuličkа na nakloněné rovině*

Současná verze reálného systému na Obr. 4.17 plně odpovídá 3D CAD modelu - Obr. 4.4. Veškeré součásti až do těch nejmenších detailů, jejich výroba a mecha-

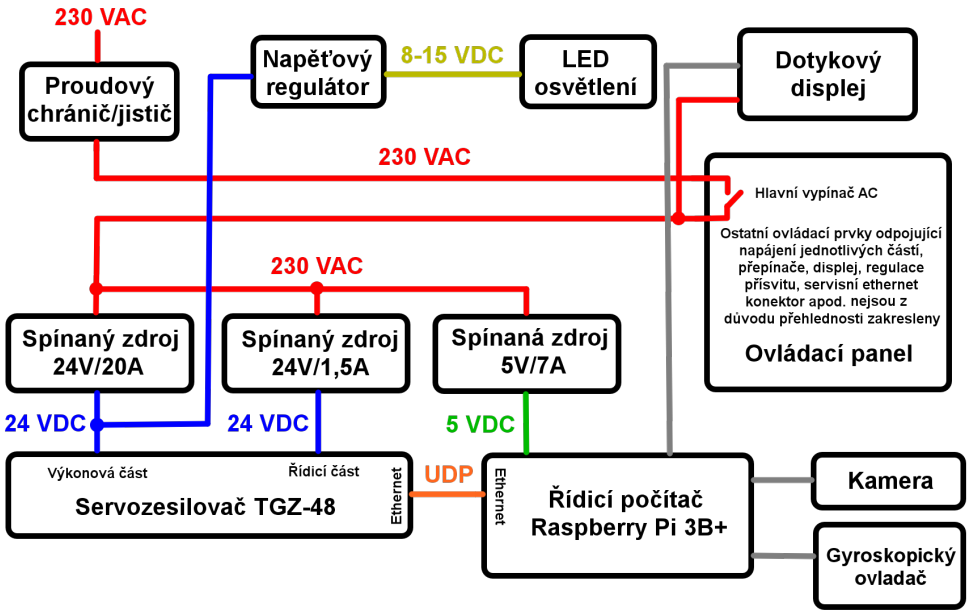
Obr. 4.18 Deska s elektrickým vybavením a ovládacím panelem

nická kompletace je autorova práce a spolu s návrhem a zprovozněním zaujímá značnou a jednoznačně nezanedbatelnou část této práce. Z tohoto důvodu je jí také věnovaný odpovídající rozsah. Výroba základních stavebních části modelu byla realizována v domácích podmínkách (Obr. 4.19), použitý materiál je výhradně hliník a nerezová ocel.

Obr. 4.19 Výroba základních stavebních částí modelu

4.4.3 Zapojení/propojení

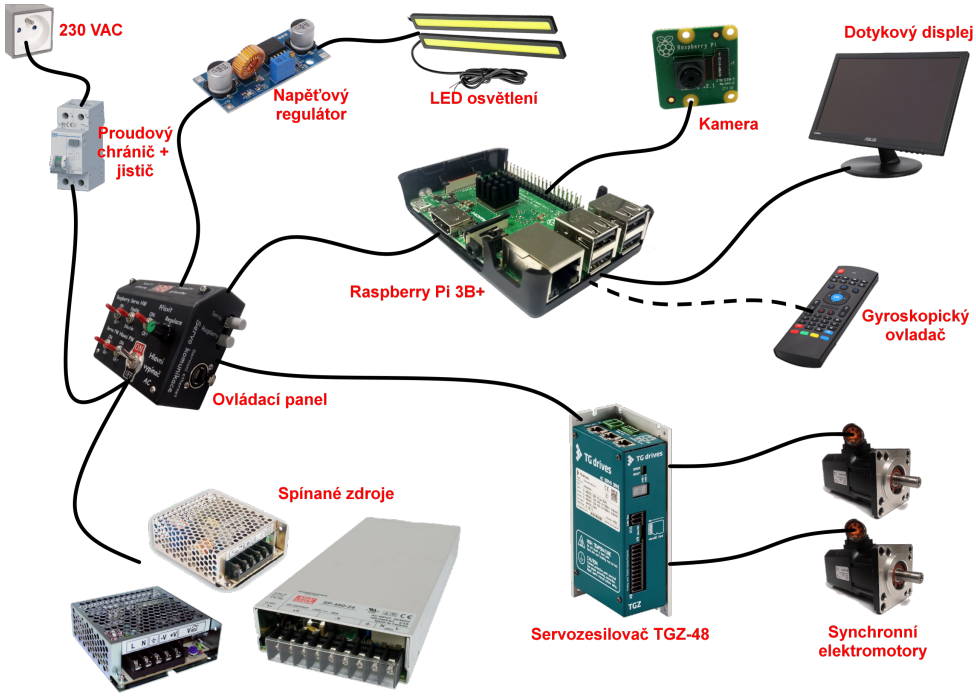
Distribuované řízení je v současnosti běžně používanou metodou využívající víceúrovňové rozdělení řešeného problému. Za nejnižší „vrstvu“ je možné v tomto případě považovat servozesilovač, nebo jinou výkonovou elektroniku zajišťující převod signálových vstupů na výkonové/energetické výstupy. Této jednotce jsou předávány informace z nadřazeného systému (řídicího počítače), na kterém může probíhat výpočet regulátorů, zpracování signálů ze senzorického sytému, může zde být implementováno uživatelské rozhraní pro parametrizaci/vizualizaci apod. S ohledem na spolehlivost je vždy doporučováno používat nezávislé/oddělené napájení jednotlivých úrovní, tedy minimálně řídicího počítače a servozesilovače. Tyto periferie jsou propojeny sběrnicí/komunikačním rozhraním s protokolem zajišťujícím požadovanou rychlost a spolehlivost. Mezi rychlostí a spolehlivostí je vždy nutné volit jakýsi kompromis, pokud už není forma komunikace pevně dána výrobcem použitých periferií.



Obr. 4.20 Zjednodušené blokové schéma zapojení desky s elektronikou

Na Obr. 4.20 je zobrazeno zjednodušené elektrické zapojení desky s elektronikou. Kvůli přehlednosti je vynechané propojení ovládacího panelu (zobrazeného

na Obr. 4.18 vpravo) se zbytkem periferií a bude popsáno dále. Model má integrovaný klasický počítačový napájecí kabel 230 VAC, připojený do konektoru na vstupu proudového chrániče a jističe v jednom dvou-modulovém provedení. Hliníková deska je uzemněna. Fázový vodič z výstupu chrániče vede do ovládacího panelu, kde je hlavní vypínač (jediný vypínač 230 VAC). Dále pokračuje fázový vodič spolu s nulovým a zemnícím vodičem do všech 3 spínaných zdrojů a dotykového displeje, kde jsou pospojovány. Dále pokračuje už pouze bezpečné napětí v rozsahu 5-24 VDC.



Obr. 4.21 Zjednodušené názorné schéma zapojení jednotlivých komponent

Výstup spínaného zdroje 24V/20A je připojený na výkonovou část servozesilovače. Tento spínaný zdroj má zapojené rovněž vzdálené logické ON/OFF ovládání na bezpečné vypnutí zdroje (na ovládacím panelu vypínač *Hlavní PW*). Spínaný zdroj 24V/1,5A je přes vypínač na ovládacím panelu zapojený do řídicí části servozesilovače, přerušením + vodiče v konektoru na pozici *+24* je ovládáno vypnutí/zapnutí silové části (na ovládacím panelu vypínač označený jako *Servo*

PW), přerušením vodiče na pozici *EN* se zase ovládá hardwarový enable/disable motorů (vypínač *Servo HW*).

Spínaný zdroj 5V/7A napájí řídicí počítač Raspberry Pi a je rovněž zapojený přes tlačítko na ovládacím panelu *Raspberry*. Servozesilovač je s řídicím počítačem propojený přes ovládací panel ethernetovým kabelem, kde bude pomocí UD protokolu probíhat vzájemná komunikace. Na ovládacím panelu je přepínač pro komunikaci se servozesilovačem buď přes servisní konektor (*Servis - Servisní ethernet*), nebo přímo pomocí Raspberry (*Raspberry*). Do řídicího počítače je připojený dále dotykový displej, kamera, gyroskopický ovladač apod.

Ze spínaného zdroje 24V/20A je také přes vypínač na ovládacím panelu (*Přísvit*) napájen napěťový regulátor ovládající intenzitu LED osvětlení. Regulátor byl doplněn externím potenciometrem pro nastavení intenzity přísvitu z ovládacího panelu (označeno jako *Regulace*), displejem pro zobrazení aktuálního nastaveného napětí (*Napětí přísvitu*) a kondenzátorem pro filtraci šumu z potenciometru (problikávání osvětlení při přenastavování). Výstup byl dále připojen na LED osvětlení a napětí bylo přizpůsobeno vhodným rezistorem na rozsah 8 - 15 VDC.

4.5 Softwarová aplikace

V této podkapitole bude popsána grafická aplikace, která je součástí hlavní a nejrozsáhlejší pasáže práce. Bude zde představeno uspořádání aplikace, rozdělení dílčích částí do jednotlivých vláken, komunikační rozhraní, GUI, zpracování obrazu z kamery pro účely sledování objektu zájmu apod.

V současnosti je grafické uživatelské rozhraní (GUI) považováno za samozřejmost. Z něj provádí koncový uživatel parametrizaci systému, diagnostiku, vidí aktuální informace o dění a má k dispozici vše, co je potřebné k ovládání celého modelu.

Možností pro tvorbu grafické aplikace je hned několik. Mezi nejznámější patří např. *.NET*, *Java*^{TM} a *Qt*. Poslední zmíněný multiplatformní open-source framework je široce používaný a v současnosti čím dál více oblíbený. Má ukázkovou dokumentaci, přehledně řešenou komunikaci mezi objekty (tzv. signály a sloty) a jednoduchou správu vícevláknových/paralelních struktur. Aplikace bude programována v C++, což je nativní jazyk Qt i IDE Qt Creatoru. Z výše uvedených důvodů a na základě osobní zkušenosti s Qt a s C++ byla volba frameworku a programovacího jazyka víceméně jasná.

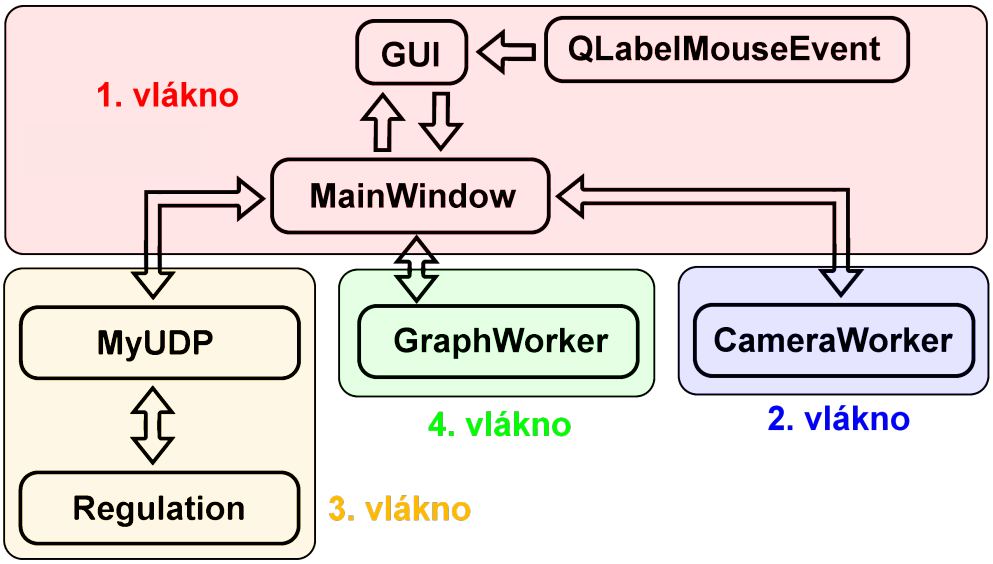
4.5.1 Struktura aplikace

Aplikace ovládající reálný systém je navržená s ohledem na zvolený řídicí počítač, senzorický systém, použitý výkonový servozesilovač a další obslužné periferie. Raspberry Pi 3B+ (řídicí počítač) je jednoznačně „nejslabším“ článkem celého řídicího systému, protože v současnosti snad neexistuje ekonomicky přívětivější varianta kompaktního počítače s operačním systémem, 4 fyzickými jádry a odpovídajícím I/O rozhraním. Při použití algoritmů zpracování obrazu, řady kinematických transformací a přepočítáváním zákona řízení (vše v reálném čase), je integrace tohoto hobby počítače do systému řízení pohybu opravdovou výzvou.

Aplikace bude muset zvládat plynulé ovládání grafického prostředí, zároveň zpracovávat obraz z kamery s maximální možnou vzorkovací frekvencí, komunikovat se servozesilovačem a jeho senzorickým systémem na frekvencích nad 1 kHz, přepočítávat regulátory v reálném čase, zvládat překreslovat regulační po-

chody a geometrické parametry, související s přepočtem pomocí kinematických transformačních matic a mnoho dalšího. Jednotlivé části aplikace se nemohou vzájemně ovlivňovat do takové míry, že by mohla např. vizualizace narušit regulační pochody, omezit rychlost komunikace se servozesilovačem apod.

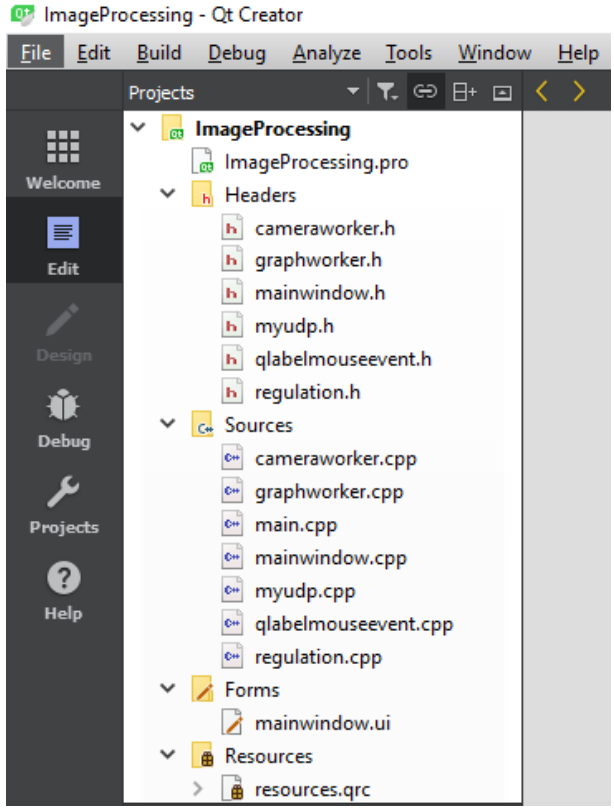
Cílem tedy nebylo vybrat výkonný počítač se systémem reálného času a grafickým čipem, který by hravě zvládal obsloužit vše potřebné i bez odladění, ale zvolit atraktivní, ekonomicky dostupnou variantu a na ní ukázat, že optimalizace kódu a rozdělení dílčích úkonů do paralelních větví dokáže razantně zredukovat očekávané výkonové požadavky na řídicí systém.



Obr. 4.22 Struktura aplikace reálného modelu

Aplikace obsahuje 6 tříd rozdělených do 4 vláken. Rodičovská třída *MainWindow* je napojená na formulář označený v Obr. 4.22 jako *GUI* a prostřednictvím událostí vyvolaných v tomto grafickém rozhraní předává pomocí signálů požadavky dále. Ve výchozím vlákně označeným číslem 1 zůstává rodičovská třída *MainWindow* spolu s grafickým rozhraním a třídou zahrnující odchytávání událostí z myši *QLabelMouseEvent* nad objektem *QLabel* - oblast vykreslování obrazu z kamery spolu s výsledky jeho zpracování. Ostatní třídy jsou přesunuty do zbylých 3 vláken podle toho, jaké úkony je výhodné od sebe oddělit. Výsledkem

rozdělení stěžejních částí do samostatných vláken poté bude fakt, že výpočetní složitost operací v jedné třídě nebude negativně ovlivňovat rychlost zpracování dílčích úkonů v třídě druhé. Vlákno s číslem 2 tedy náleží veškerým operacím s obrazem z kamery (třída *CameraWorker*), vlákno číslo 3 obstarává komunikaci se servozesilovačem a výpočet regulátorů (třídy *MyUDP* a *Regulation*) a poslední volné vlákno číslo 4 je vyhrazeno pro grafickou vizualizaci snímaných veličin, regulačních pochodů a pro následný export dat k pozdější analýze (třída *GraphWorker*).

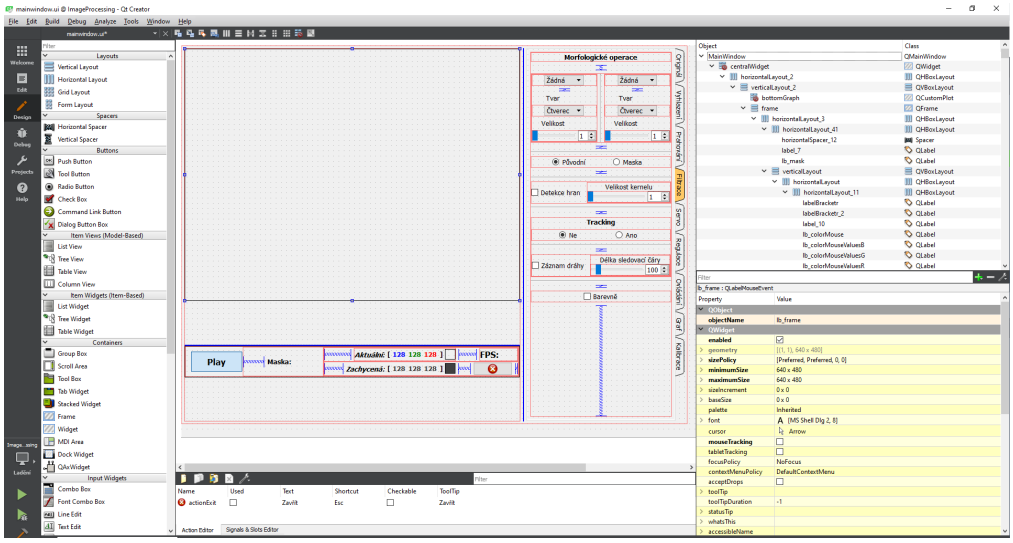


Obr. 4.23 Složení aplikace ve vývojovém prostředí Qt Creator

Cílem následujícího popisu jednotlivých tříd není provést detailní rozbor všech částí, ale rámcově nastínit, jaká je jejich hlavní úloha. Nebudou zde tedy žádné ukázky zdrojového kódu s popisem apod. - celý program obsahuje přes 20 000 řádků kódu a tato práce se nezabývá tvorbou grafické aplikace, ač je pro systém

nezbytná a časově velice náročná. Detailní komentář je k dispozici v přiloženém zdrojovém kódu a také v dokumentaci. Struktura aplikace ve vývojovém prostředí QT Creator je zobrazeno na Obr. 4.23.

Aplikace je pracovně pojmenována jako *ImageProcessing*, stejnojmenný soubor s příponou *.pro* slouží k popisu použitých zdrojových souborů a postupu vytvoření spustitelného programu, zahrnuje tedy mimo jiného cesty ke knihovnám a s tím související volbu, na jakém operačním systému probíhá překlad aplikace. Soubor *resources.qrc* obsahuje uloženou sadu ikon použitelných v aplikaci, *main.cpp* slouží pouze ke spuštění aplikace, tedy vytvoření (příp. nastavení) a zobrazení hlavního widgetu. Nejzajímavější je zde soubor *Mainwindow.ui*. Ten obsahuje XML kód, který nese veškeré informace o navržené grafické aplikaci (hlavního widgetu). GUI se nevytváří v textovém editoru, ale využívá se zabudovaného QT Designeru - Obr. 4.24. Toto prostředí je založeno na přetahování ovládacích prvků do hlavního okna a jejich tabulkové parametrizaci - v současnosti standardní postup.



Obr. 4.24 Ukázka tvorby GUI v Qt Designeru - součást Qt Creator

Zbývající hlavičkové a zdrojové soubory odpovídají třídám ve schématu na Obr. 4.22 a budou popsány níže.

MainWindow

Jak již bylo řečeno, tato třída je rodičovská, zprostředkovává veškeré propojení mezi jejími potomky a je přímo propojená s grafickým rozhraním, z kterého odchytává události ovládacích prvků. Probíhá zde také přesun ostatních tříd do vláken, inicializace po spuštění, uložení/načtení nastavených parametrů aplikace atd. Třída *MainWindow* je tedy zjednodušeně řečeno hlavním spojovacím uzlem mezi jednotlivými třídami.

GUI

Takto je pouze v blokovém schématu na Obr. 4.22 pojmenován soubor pro uživatelské rozhraní *mainwindow.ui*, ve kterém je vytvářený grafický vzhled a jsou voleny ovládací/zobrazovací prvky aplikace. Tento soubor se při kompilaci převede nástrojem *uic* na standardní hlavičkový C++ soubor. V tomto případě z *mainwindow.ui* vznikne *ui_mainwindow.h*, kde bude implementovaný navržený widget. Poté se už jen v *mainwindow.cpp* vytvoří instance tohoto widgetu a ručně dopíše, co je potřeba.

QLabelMouseEvent

Tato třída je navázána na objekt *QLabel*, do kterého se vykresluje obraz z kamery s výsledky zpracování obrazu a regulace. Třída zastřešuje odchytávání událostí tlačítek myši a předává její pozici v obrazu. Tím umožňuje např. nastavení barvy pro prahování, označení oblasti zájmu pro práci s obrazem, ovládání jednotlivých regulačních smyček modelu kurzorem myši apod. Zejména možnost označení požadovaného barevného odstínu sledovaného objektu je uživatelsky velice přínosným prvkem aplikace.

CameraWorker

Zde již bylo vytvořeno paralelní vlákno, které je vyhrazené pro práci s obrazem. Algoritmy zpracování obrazu jsou výpočetně dosti náročné a na rychlosti jejich vykonání závisí kvalita regulace nejvyšší smyčky - řízení polohy kuličky na nakloněné rovině. Z tohoto důvodu nesmí být, pokud možno, nijak narušována jejich plynulost. Instance třídy *CameraWorker* je tedy po vytvoření přesunuta do ji-

ného vlákna, než je vlákno, ve kterém běží grafické rozhraní. Tato třída obsahuje stěžejní část celé aplikace a bude rozebrána v samostatných podkapitolách.

Ve zkratce zde probíhá nastavení kamery (jas, kontrast, saturace, ...), sejmutí originálního snímku z kamery a jeho úprava podle zvolených kritérií (výřez oblasti zájmu, vyhlazení), následuje zpracování obrazu, tedy výběr objektu zájmu (prahování, odstranění šumu, detekce hran, potlačení nežádoucích objektů) a nakonec zaměření se na řízený objekt a jeho sledování. Dále třída obsahuje kinematické transformace pro přepočet mezi 2D obrazem z kamery a 3D prostorem modelu (přepočet ohraničení nakloněné roviny na základě úhlu natočení motorů, odstranění lichoběžníkového zkreslení kamery, kalibrační procedury při změně polohy a orientace kamery, výpočet skutečné polohy kuličky na nakloněné rovině), zjištění žádaného barevného odstínu na pozici kurzoru myši včetně přepočtu na libovolně ořezanou oblast zájmu, veškeré vykreslování do obrazu a mnoho dalšího.

MyUDP

Třída, pro kterou je vyhrazeno další vlákno a do kterého je její instance po vytvoření ihned přesunuta. Už podle názvu je patrné, že se stará o komunikaci mezi řídicím počítačem a servozesilovačem po UD protokolu. Zahrnuje tedy vytvoření a nastavení spojení, připojení k servozesilovači a kontrolu, jestli je spojení aktivní, časovače ošetřující definované intervaly čekání na datagram, obnovení spojení, periodicitu odesílání požadavků atd., je zde definována požadovaná struktura datagramu s příkazy pro servozesilovač, odesílání a přijímání datagramů, převod příchozích dat do požadované podoby, předávání proměnných pro grafické vykreslení apod. Probíhá zde také komunikace mezi třídou *Regulation*, které jsou předávány stavové proměnné pro výpočet regulátorů.

Regulation

Běží ve stejném vláknu jako instance třídy *MyUDP*, stará se o volbu a výpočet regulátorů, generování žádaných trajektorií, obsahuje režimy automatického/-manuálního řízení a tedy i možnost napojení se na libovolnou regulační smyčku. Jsou zde naprogramovány diskrétní regulátory s vhodnými náhradami a úpra-

vami, které vylepšují kvalitu regulace.

GraphWorker

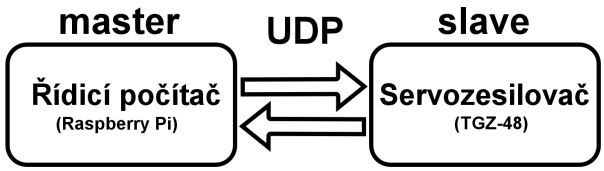
Pro tuto třídu je vyhrazené poslední volné vlákno. Jedná se o soubor funkcí pro vykreslení požadovaných veličin do grafů a jejich následný export pro potřeby analýzy v odpovídajících programech. Jsou zde k dispozici bohaté možnosti parametrizace hlavního grafu, průběžný zjednodušený graf pro rychlý náhled dostupný na všech záložkách aplikace, možnost odchycení předem definovaného úseku (triggering) apod.

4.5.2 Volba vzájemné komunikace

Ve většině případů je forma komunikace pevně daná výrobcem alespoň jedné z periferií - často se jedná o servozesilovač, který nemá takové možnosti, jako řídicí počítač. Komunikace přes rozhraní Ethernet je stále více podporována, a to také díky množství použitelných protokolů, ze kterých je možné si s ohledem na rychlost a spolehlivost přenosu vybrat ten vyhovující.

V řízení pohybu s vysokou dynamikou je důležitým faktorem zejména rychlost reakce na vyvolanou/vyžádanou změnu stavu, proto se na rozdíl od široce používaného TCP/IP volí častěji UDP. Úspěšnost doručení dat se u UDP nijak nekontroluje, o to je ale rychlejší. V případě čekání na úspěšné doručení již neaktuálních dat je u řízení pohybu lepší inkriminovaný datagram zahodit a v dalším kroku použít datagram aktuální.

V této práci bylo pro komunikaci použito rozhraní Ethernet s UDP. Tato volba je dobrým kompromisem z výše uvedených důvodů a také je podmíněná použitým servozesilovačem. Při použití real-time operačního systému se softwarem poskytovaným výrobcem by bylo možné použít i protokol EtherCat, avšak aplikace (software) výrobce je uzavřený a není tedy možné jej modifikovat (požadavek na vlastní GUI rozhraní). Do servozesilovače je sice částečně možné doprogramovat vlastní část řídicího kódu, jsou zde však značně omezené možnosti a není možné připojit kameru ani další periferie. I přesto poskytuje zvolený servozesilovač oproti konkurenci nadstandardní možnosti parametrizace spolu s jednoduchým a přehledným komunikačním protokolem.



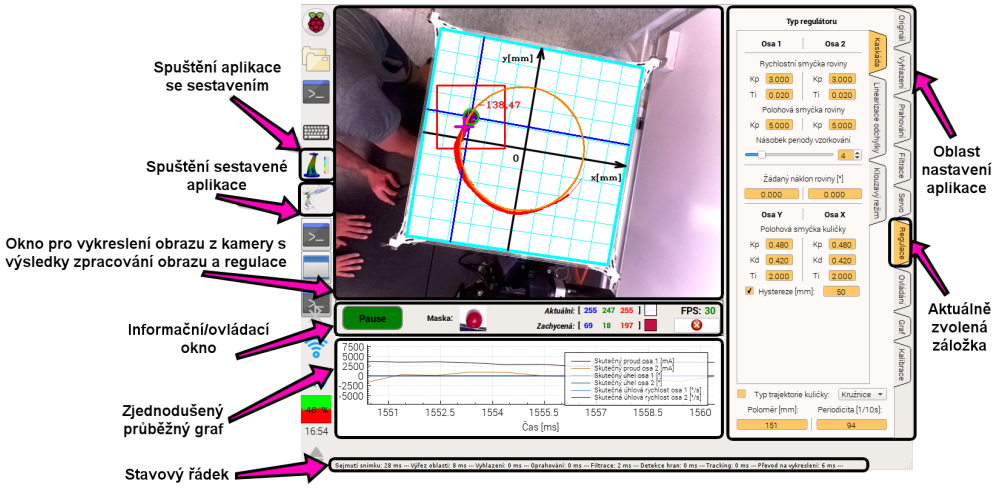
Obr. 4.25 Blokové schéma komunikace mezi řídicím počítačem a servozesilovačem

Servozesilovač lze pomocí UDP parametrizovat, monitorovat i řídit. IP adresa servozesilovače je statická, všechny kontrolní součty (IP, UDP) musí být platné, jinak bude paket ignorován. Servozesilovač je v komunikaci slave (jen odpovídá na dotazy). Na každý přijatý paket ve správném formátu odesílá paket s odpovědí. Master musí před odesláním dalšího dotazu vždy čekat na odpověď od slave. Pokud master nepřijme odpověď v časovém limitu, může se po dané době pokusit o další dotaz. První dva byte datové části náleží identifikaci, následující představují vlastní příkazy (buď zápis parametrů, nebo jejich čtení) [66].

Rychlost komunikace určuje v podstatě periodu vzorkování nejrychlejší smyčky (rychlostní v případě P(I)(D) kaskády), protože v každém odeslaném datagramu je obsažen příkaz pro přepis žádané hodnoty proudů. Při zápisu žádaných hodnot a dalších funkčních parametrů se zároveň čtou skutečné hodnoty proudů, rychlostí, poloh apod. Tyto proměnné jsou dále použity do zpětné vazby při regulaci, pro monitorování, grafické vykreslení atd.

4.5.3 Popis uživatelské části

Aplikace je složená z dvou hlavních částí, kterými jsou rozbalovací oblast pro nastavení celého systému a okno pro vykreslení obrazu z kamery, výsledků zpracování obrazu, regulace a ostatních pomocných objektů. Pod vykreslovacím oknem je oblast se základními informacemi a ovládacími prvky, pod ní se nachází zjednodušený průběžný graf a ve spodní části aplikace je k dispozici stavový řádek s průměrnými časy provedení jednotlivých kroků zpracování obrazu. Velká část je věnována nastavení, souvisejícím se zpracováním obrazu z kamery, protože je nutné vždy provést kontrolu výsledků jednotlivých operací s obrazem, které na sebe navazují (z tohoto důvodu jsou některé záložky téměř prázdné).

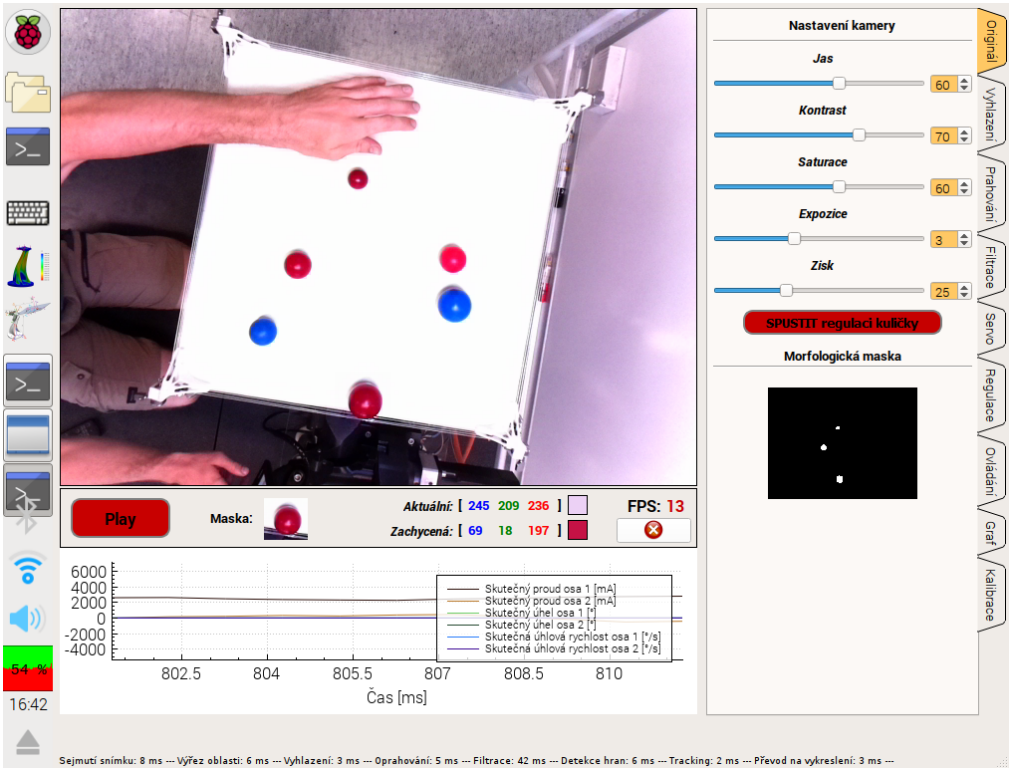


Obr. 4.26 Popis částí grafického rozhraní aplikace

Na Obr. 4.26 je ve vykreslovacím okně zobrazena část s automatickou regulací kuličky po požadované dráze. Tyrkysovou barvou je označena hranice nakloněné roviny s šachovnicí o velikosti 5 cm - vše je průběžně přepočítáváno v závislosti na náklonu roviny, umístění a orientaci kamery a dalších geometrických parametrech. Modrým záměrným křížem je zaznamenávána skutečná poloha kuličky, fialovým křížkem zase aktuální žádaná poloha kuličky. Oranžový průběh (v tomto případě kružnice) je žádaná trajektorie pohybu kuličky, červený rámeček zobrazuje vybranou oblast zájmu, v které probíhá zpracování obrazu a ztenčující-se dráha stejné barvy naznačuje historii pohybu kuličky.

Tlačítko *Pause/Play* pod oknem spouští a zastavuje vyčítání obrazu z kamery, ikona s červeným křížkem zavírá aplikaci. Identifikace a sledování cíle je založeno na výběru jeho barevného odstínu, je zde ale připraveno i sledování objektu na základě výběru masky. Vektor s názvem *Aktuální* zobrazuje barvu pixelu na pozici kurzoru myši, vektor *Zachycená* zase informuje o zachycené barvě objektu, který je sledován. FPS udává průměrnou vzorkovací frekvenci, s kterou je obraz z kamery zpracováván.

Záložka *Originál*:



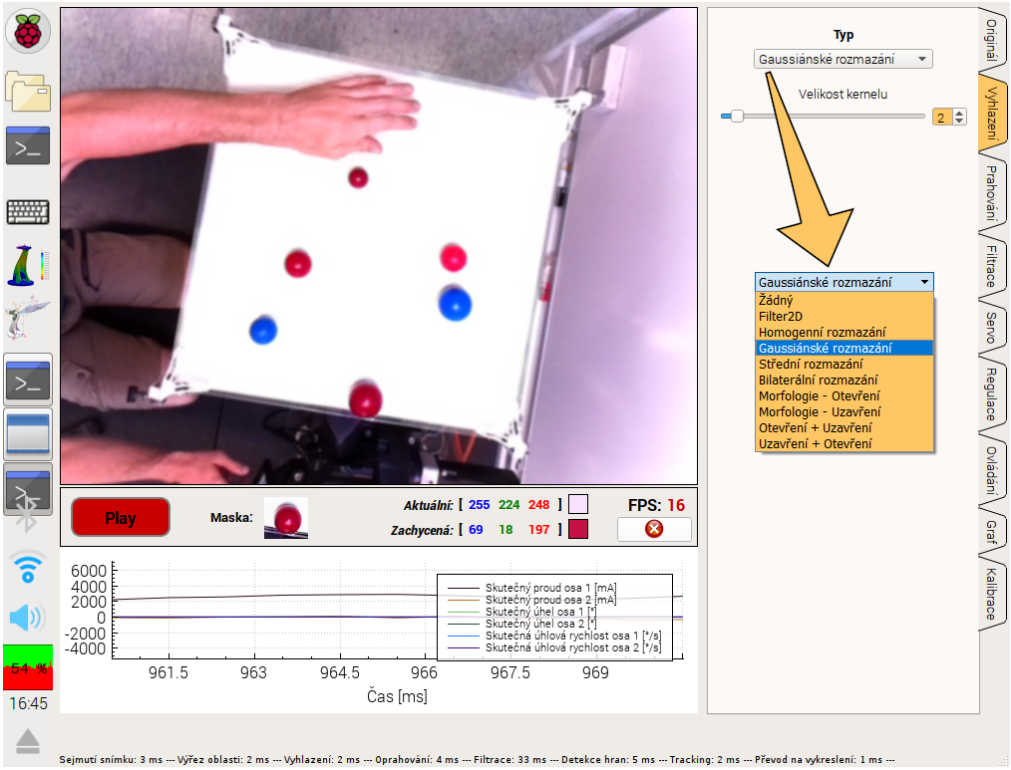
Obr. 4.27 Obsah záložky *Originál*

V této záložce se nastavují všechny potřebné parametry videa. Mezi základní 2 veličiny patří *Jas* a *Kontrast*. Dalším užitečným parametrem, zejména při práci s barevným odstínem, je *Saturace*. Je zde také možné nastavit čas expozice a zisk, tedy zesílení jasu obrazu. Tlačítko *SPUSTIT regulaci kuličky* svým názvem plně vystihuje svou funkci, *Morfologická maska* zobrazuje zmenšeninu výsledného snímku po zpracování obrazu, ještě před procedurou výběru a následného sledování kýženého objektu.

Záložka *Vyhlazení*:

Vyhlazením se z obrazu částečně odstaní šum, což má pozitivní vliv na další práci s ním. Existuje mnoho postupů, jak obraz vyhladit - od jednoduchých rychlých metod, které obraz pouze rozostří, až po metody zachovávající ostré přechody

bez vedlejších efektů. Zde je volba kompromisu mezi kvalitou výsledného snímku a rychlostí zpracování opravdu klíčová. Je implementováno celkem 9 různých filtrů a kombinací základních morfologických operací, které byly otestované a jsou použitelné pro práci v reálném čase.

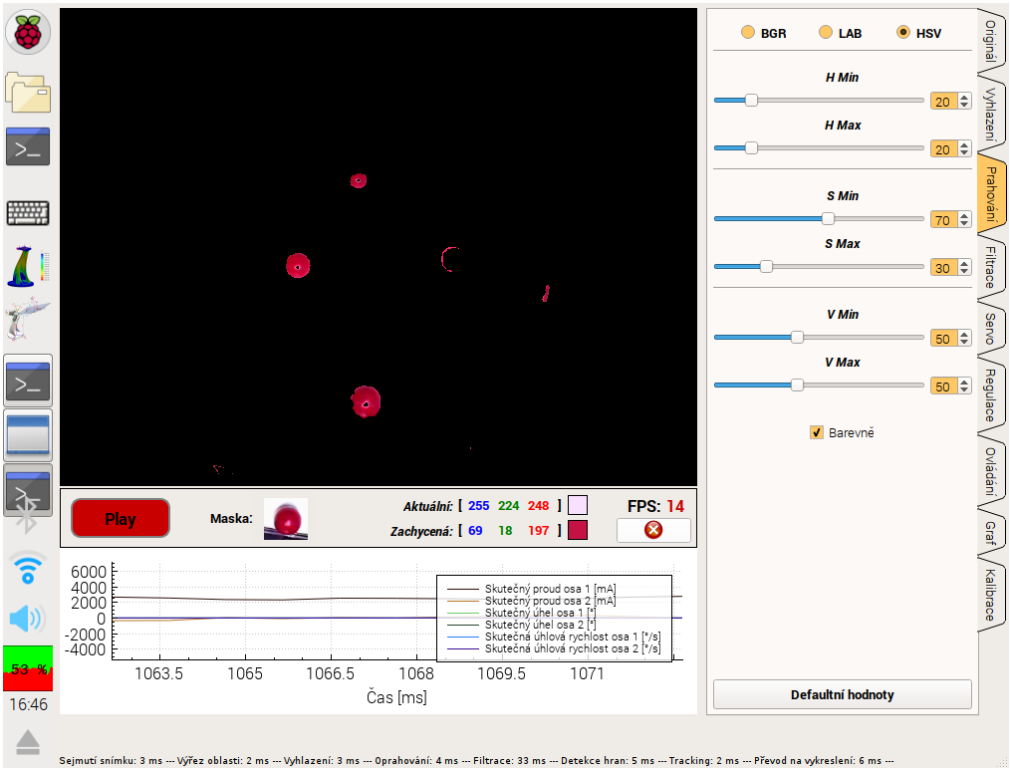


Obr. 4.28 Obsah záložky *Vyhlazení*

Parametr *Velikost kernelu* určuje zjednodušeně řečeno míru vyhlazení (potažmo rozmazání) obrazu. Se zvětšováním tohoto parametru rostou výpočetní nároky jednotlivých metod, protože se zvětšuje množství operací k provedení (kernel určuje velikost konvoluční masky).

Záložka *Prahování*:

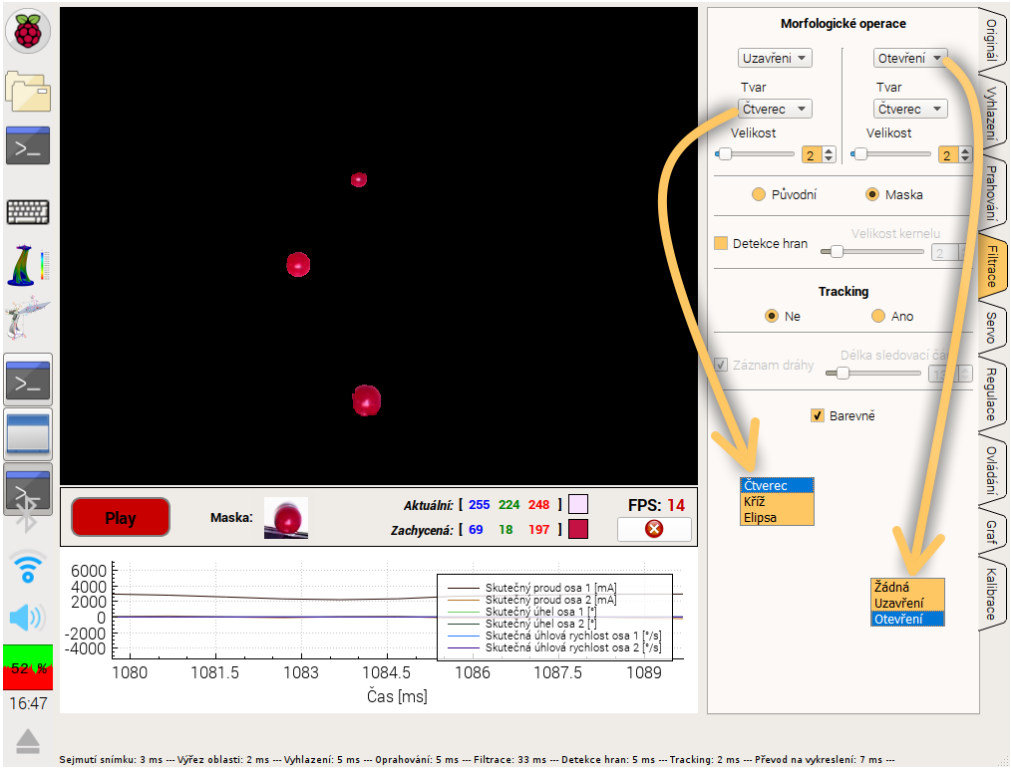
V této záložce probíhá nastavení prahové hranice mezi ponechanou barvou a barvou určenou k odstranění. Správné nastavení rozsahů a volba vhodné barevné palety zásadně ovlivňuje výsledek celého zpracování obrazu a rozhoduje



Obr. 4.29 Obsah záložky *Prahování*

o robustnosti této metody. Je možné volit mezi 3 barevnými paletami, kterými jsou *BGR*, *LAB* a *HSV*. Všechny modely stále pracují s mícháním 3 základních složek barev (modrá, zelená a červená), snímek z kamery je totiž uložený nativně v paletě BGR, takže není ani jiná možnost. Rozdíl mezi těmito modely je pouze ve vyjádření daného barevného odstínu jinými parametry, pomocí nichž je barva z pohledu člověka více intuitivní a lépe rozpoznatelná. Nastavované hodnoty vyjadřují odchýlení od zvolené barvy v jednotlivých složkách vybrané barevné palety. Určují tedy míru tolerance od dané složky, s jakou bude barva ještě ponechána. Zaškrtávací políčko *Barevně* stanoví, jestli se bude výsledný snímek zobrazovat černobíle (výsledkem prahování je vždy černobílý), nebo se vybrané oblasti vyřežou z původního snímku, viz Obr. 4.29. Tlačítko *Defaultní hodnoty* nastaví prahovací meze na doporučené, od kterých poté provádí uživatel korekturu podle světelných podmínek a barevného odstínu sledovaného objektu.

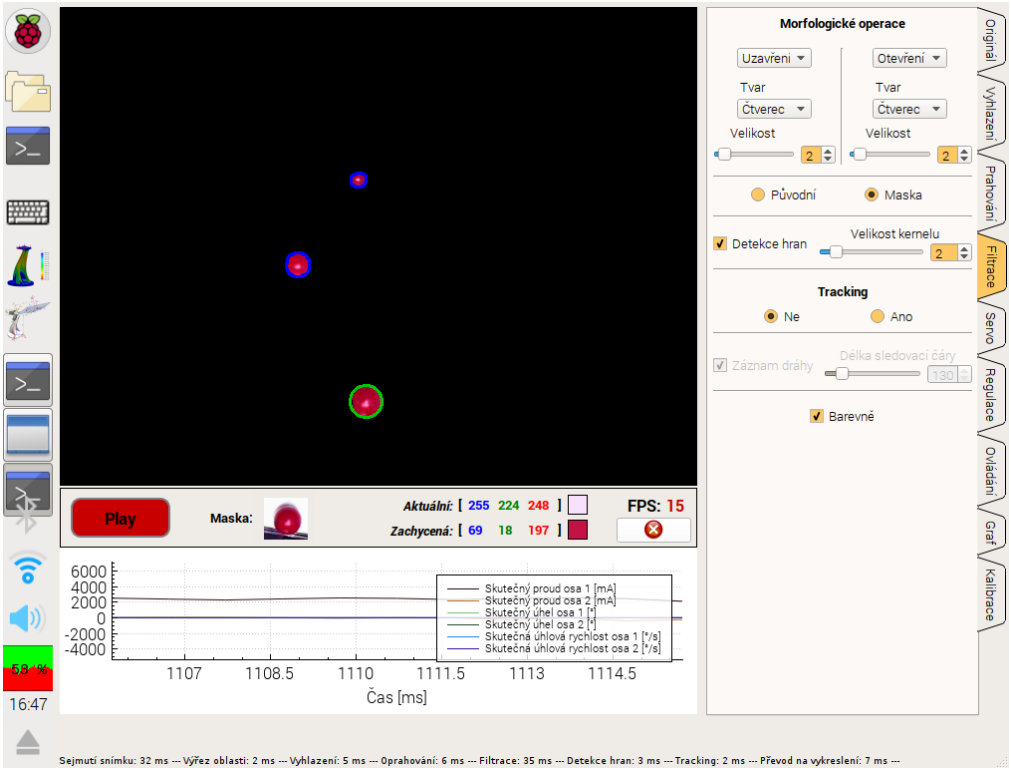
Záložka *Filtrace*:



Obr. 4.30 Obsah záložky *Filtrace* - morfologické operace

Poslední záložkou zabývající se zpracováním obrazu z kamery je záložka *Filtrace*. Je zde více na sebe navazujících částí, z nichž první je aplikace morfologických operací na oprahovaný snímek. Tyto matematické operace vycházejí z teorie množin a jejich úkolem je odstranit z obrazu zbylý šum. Za šum lze v tomto případě považovat i nechtěné, ne-zcela odstraněné objekty - vše záleží na nastavení velikosti kernelu a typu operace. V nabídce je možné volit mezi 2 typy složených morfologických operací, a to *Uzavření* a *Otevření*. Navazují na sebe a je možné je také úplně vynechat volbou *Žádná*. Uspořádání a typ operací byl vybrán na základě experimentálních ověření robustnosti a časové náročnosti jednotlivých metod. Po zvolení typu morfologické operace se volí tvar a velikost jádra/kernelu - význam obdobný, jako u záložky *Vyhlazení*. Přepínáním mezi *Původní* a *Maska* je vybráno, jestli se bude výstup po morfologických operacích

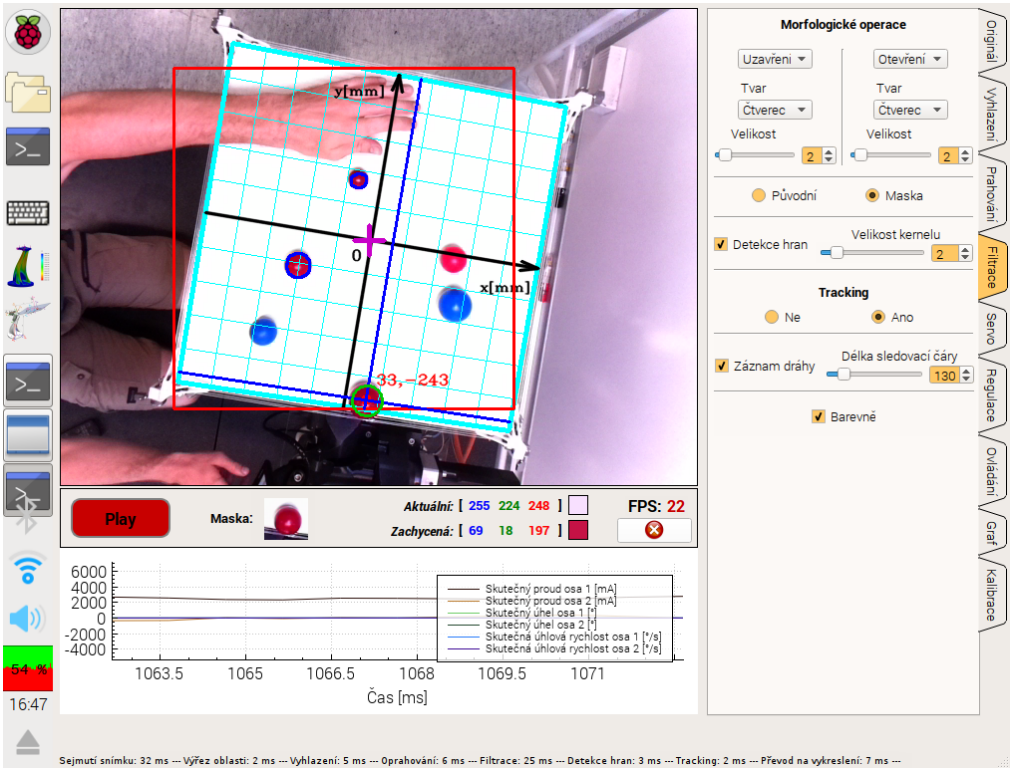
aplikovat na snímek po oprahování (*Původní*), nebo jestli se využije výsledek morfologických operací jako maska a udělá se průnik s originálním snímkem z kamery (*Maska*) - použito v Obr. 4.30.



Obr. 4.31 Obsah záložky *Filtrace* - detekce hran

Druhou částí záložky *Filtrace* je detekce hran ponechaných objektů (zaškrtávací políčko *Detekce hran*) a výběr jednoho z nich. V tomto případě je objektem zájmu volen ten největší (největší nalezený průměr kruhového ohraničení). Posuvníkem s názvem *Velikost kernelu* se volí tloušťka čáry ohraničení. Zelenou barvou (Obr. 4.31) je ohraničený objekt zájmu určený dále ke sledování, modře pak zbývající potenciální objekty zájmu.

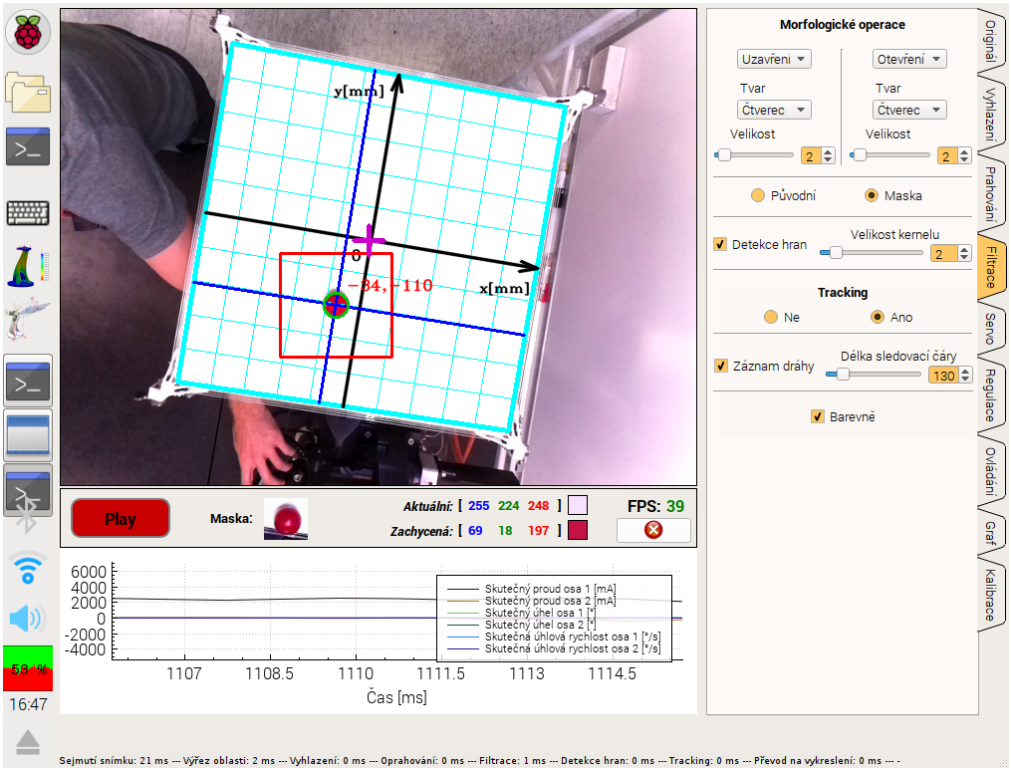
Poslední částí je sledování zeleně označeného objektu zájmu - přepínání mezi *Ano/Ne* ve skupině *Tracking*. Dále je zde volba *Záznam dráhy*, při jejíž využití se do obrazu vykresluje historie polohy kuličky na nakloněné rovině (lze vidět při regulaci např. na Obr. 4.35). Posuvník *Délka sledovací čáry* udává, jak velké je



Obr. 4.32 Obsah záložky *Filtrace* - sledování objektu - krok 1

pole vzorků pro ukládání historie pozic kuličky, tedy jak dlouhá je vykreslovaná čára za kuličkou. Zaškrtávací políčko „Barevně“ udává, obdobně jako u předchozích záložek, jak bude zobrazen výsledek - buď bude zobrazena černobílá maska, nebo barevný výřez z originálního snímku.

Na závěr je třeba podotknout, že snímek z kamery na Obr. 4.32 se v této záložce nezobrazí, ale bude vypadat stejně jako na Obr. 4.31 s tím rozdílem, že se bude oblast zájmu posunovat podle pohybu sledovaného objektu. Snímek z kamery na Obr. 4.32 je výsledný snímek dostupný v ostatních záložkách mimo záložek řešících zpracování obrazu a je zde pro demonstraci vyznačené oblasti zájmu - červený rámeček. Snímek je již ořezaný podle detekovaného ohraničení nakloněné roviny, nezpracovává se tedy už celý, ale pouze jeho část (oproti předchozím kroků, ve kterých byl ještě zpracovávám celý snímek). Je zde zaznamenatelný nárůst vzorkovací frekvence, viz parametr *FPS*).



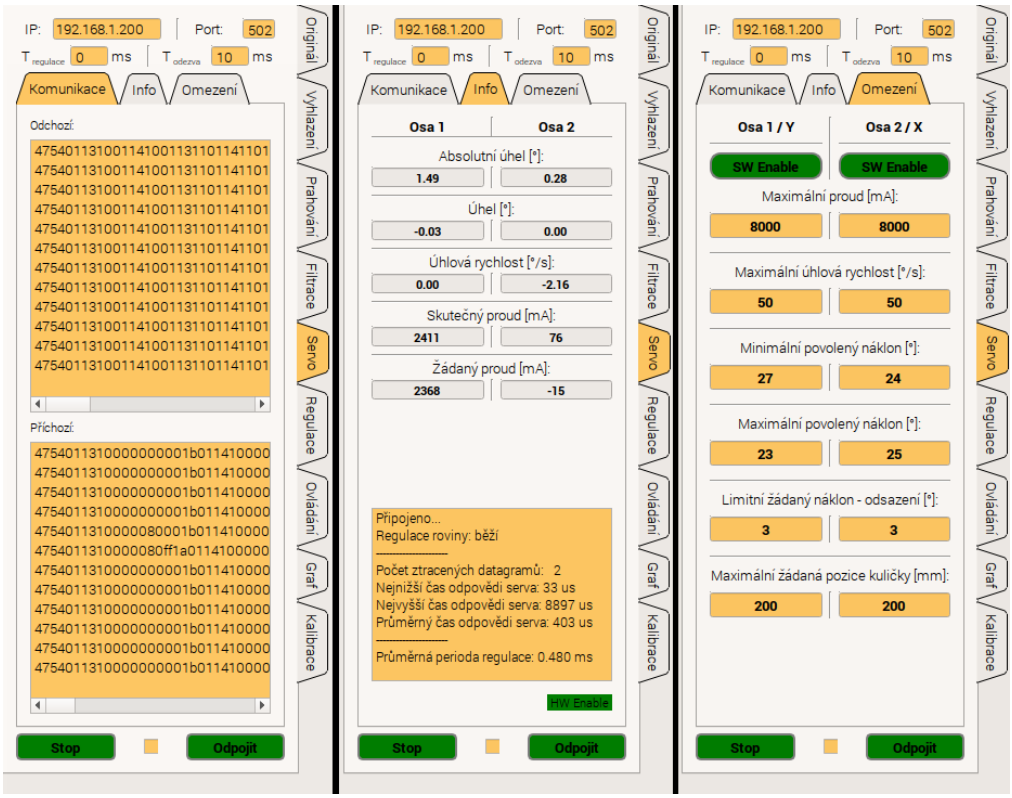
Obr. 4.33 Obsah záložky *Filtrace* - sledování objektu - krok 2

Zásadní krok pro výrazné omezení požadavků na výpočetní výkon je zobrazen na Obr. 4.33. Zde už je označena menší oblast zájmu, která se v závislosti na pohybu kuličky posouvá. Na základě tohoto zmenšení bylo dosaženo téměř dvojnásobného nárůstu vzorkovací frekvence, s kterou se stíhá obraz přepočítávat.

Záložka *Servo*:

V této záložce probíhá nastavení připojení a parametrů servozesilovače. Obsahuje nastavení IP adresy a portu servozesilovače (*IP* a *Port*), dále nastavení periody regulace, tedy času odesílání požadavku na servozesilovač ($T_{regulace}$) a maximální odezvu, kdy řídicí počítač čeká na odpověď servozesilovače (T_{odezva}). Tlačítko *Připojit/Odpojit* navazuje vzájemnou komunikaci mezi řídicím počítačem a servozesilovačem, tlačítko *Start/Stop* spouští/zastavuje regulaci nakloněné roviny a zaškrtávací políčko mezi nimi povoluje/zakazuje výpis nativního tvaru

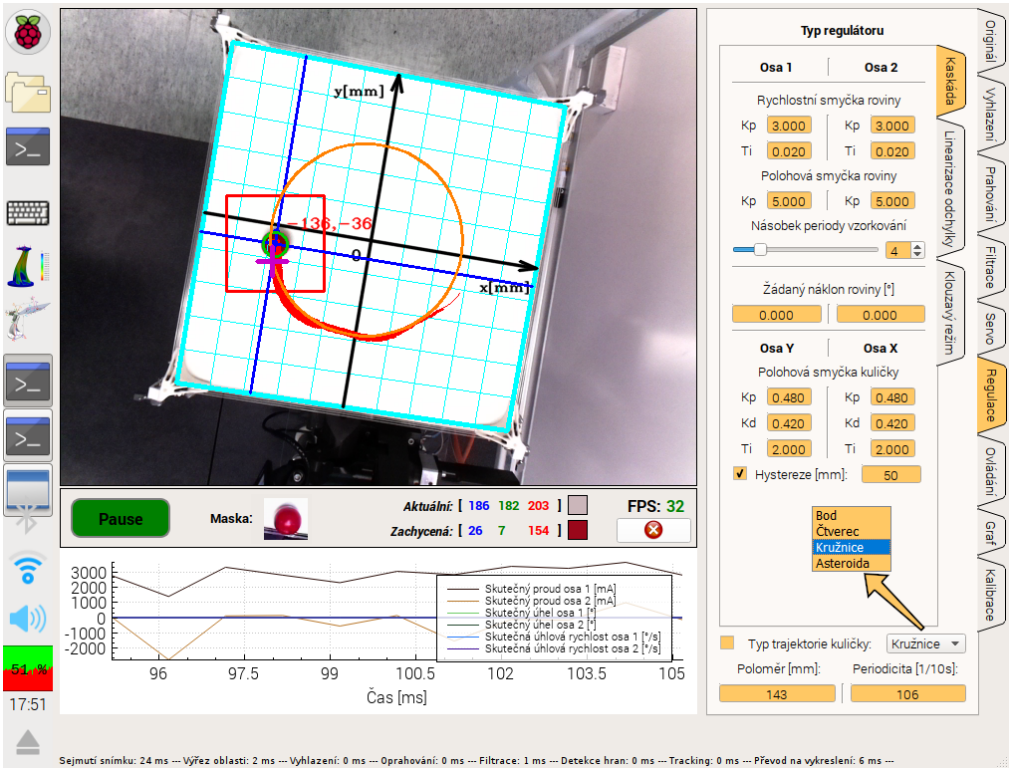
příchozí a odchozí zprávy do pod-záložky *Komunikace*. Další pod-záložka *Info* obsahuje informační okno a část s důležitými aktuálními hodnotami ze servozesilovače. Poslední pod-záložka *Omezení* se zaobírá nastavením mezí stavových veličin a připojením/odpojením aktuátorů jednotlivých os.



Obr. 4.34 Obsah záložky *Servo*

Záložka *Regulace*:

Zde je možné nastavit parametry regulátorů pro zvolené typy regulace. Na Obr. 4.35 je ukázka nastavení pro kaskádní regulační strukturu. Popis jednotlivých parametrů není potřeba uvádět, protože přímo odpovídají konstantám regulátorů. Položka s názvem *Násobek periody vzorkování* nastavuje násobek vzorkovací periody polohové smyčky oproti smyčce rychlostní ($T_{regulace}$ v záložce *Servo*). Proměnná *Hystereze* zase určuje, jestli bude aplikovaná integrační složka v polohové



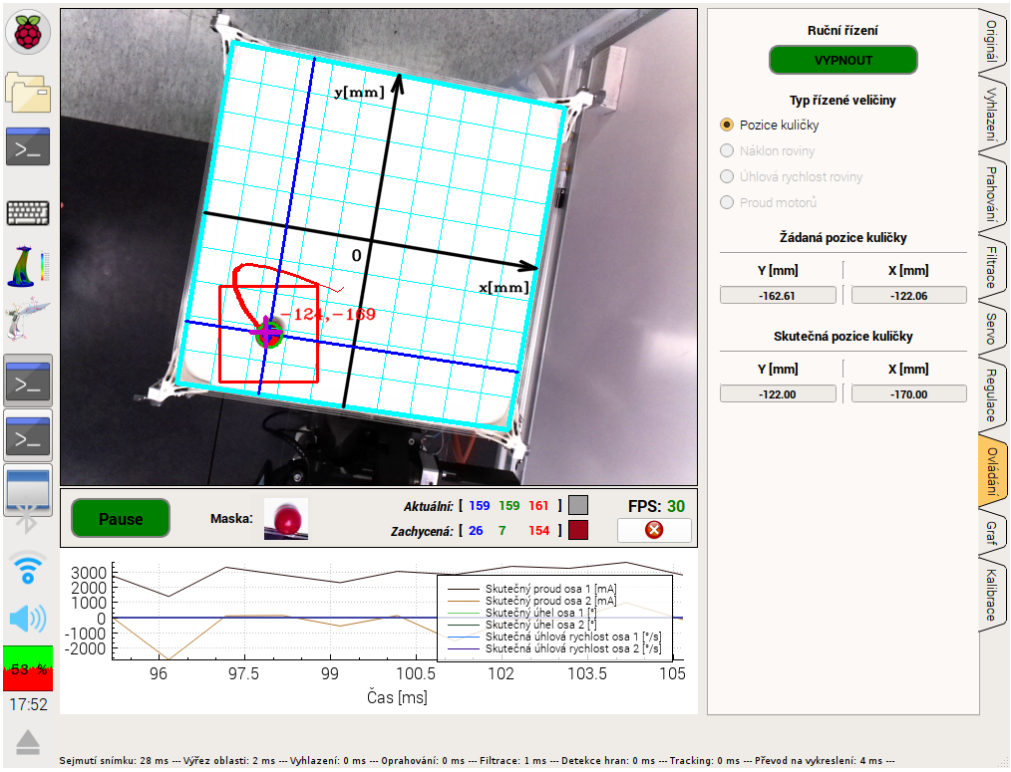
Obr. 4.35 Obsah záložky *Regulace*

smyčce kuličky a při jaké hodnotě regulační odchylky se bude připínat.

Ve spodní části záložky se nachází volba požadované trajektorie, kterou má kulička kopírovat. Jsou zde 4 možnosti, při výběru *Bod* se nastavuje žádaná [x, y] pozice kuličky, ostatní už jsou trajektorie ve tvaru *Čtverec*, *Kružnice* a *Asteroida*. Žádanou hodnotou je opět bod, jehož pozice ale není konstantní - pohybuje se pro zvolené trajektorii s poloměrem *Poloměr* a mění se v čase s periodou $Periodicita/10$.

Záložka *Ovládání*:

Tato záložka přidává možnost ručního řízení celého systému. Hodnoty z myši/gyroskopického ovladače vstupují do regulační struktury jako žádané hodnoty zvolených regulačních smyček. Jsou zde zahrnuty všechny smyčky kaskádní regulace, tedy proudová, rychlostní a polohová pro řízení nakloněné roviny a polohová

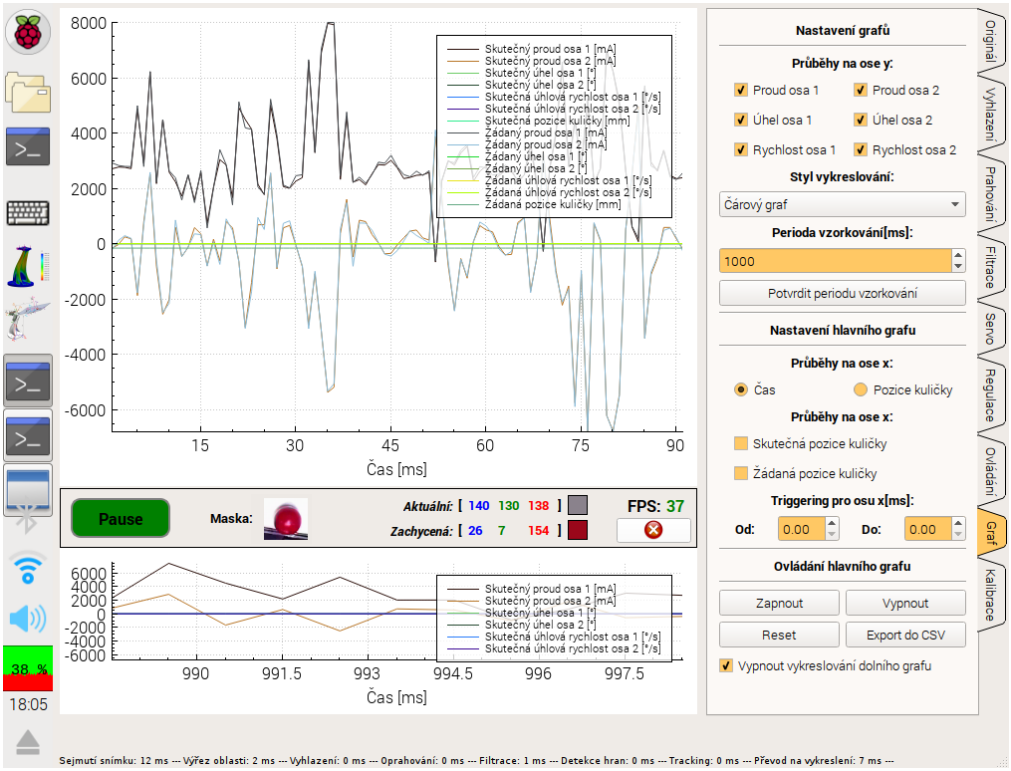


Obr. 4.36 Obsah záložky *Ovládání*

smyčka řízení pozice kuličky na nakloněné rovině. Všechny volitelné možnosti jsou v této záložce pojmenovány jasně a výstižně, není potřeba je proto dále popisovat. Po stisknutí tlačítka *ZAPNOUT/VYPNOUT* se rovina vždy stabilizuje do výchozí horizontální polohy, z které ruční řízení započíná.

Záložka *Graf*:

Zde si může nechat uživatel vykreslit všechny požadované veličiny v přehledném grafu. Je možnost zobrazit průběhy jak v časové závislosti, tak v Euklidovském prostoru pro x-y závislost. Podle charakteru zobrazované veličiny je nastavitelný interval spuštění záznamu dat (tzv. triggering) a perioda vzorkování. Charakteristiky je možné vykreslit ve 3 různých režimech, a to jako impulzy, schodovitý průběh a lineárně interpolovanou křivku. Zaznamenávaná data se mohou také ukládat do paměti a následně do *.csv* souboru, jejich záznam je možné manuálně



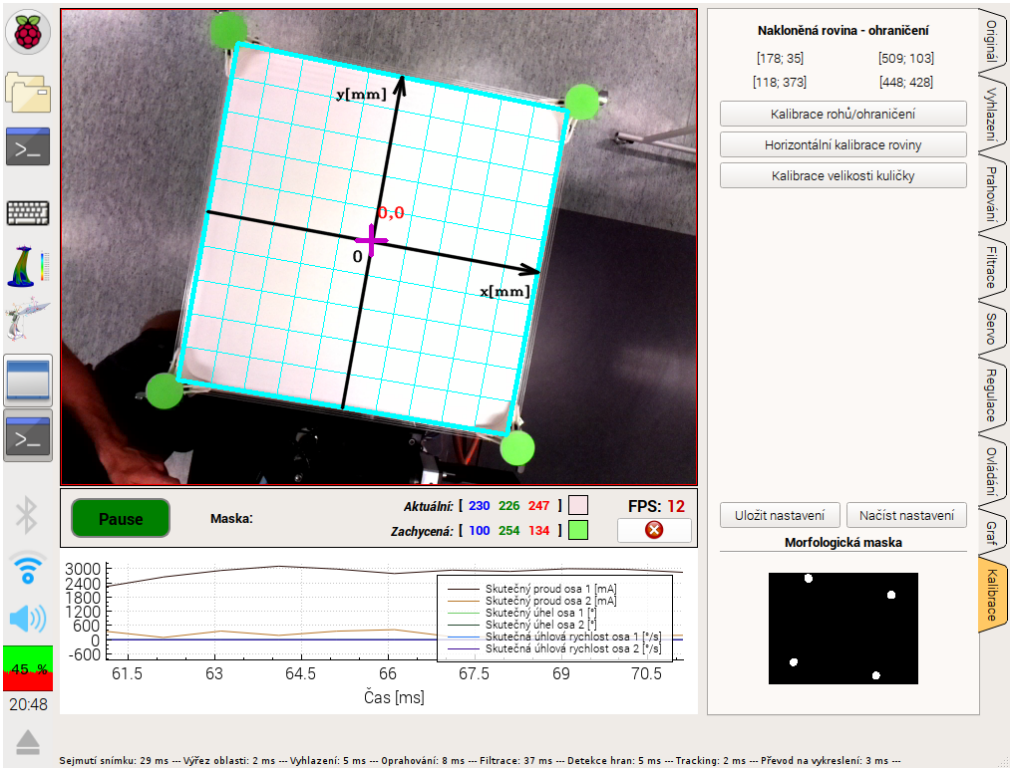
Obr. 4.37 Obsah záložky *Graf*

spustit, zastavit, nebo vymazat naměřené data z paměti *Reset* tlačítkem.

Záložka *Kalibrace*:

Poslední ze záložek se týká kalibrace. Kalibrace nakloněné roviny (*Kalibrace rohů/ohraničení*) je klíčová, protože se od ní odvíjí veškeré kinematické transformace a geometrické přepočty. Jedná se o nalezení rohů nakloněné roviny pomocí kalibračních obrazců, které se poté mohou odstranit (viz předchozí obrázky uživatelského rozhraní). Ze známého odsazení kalibračních obrazců (zelené kruhové objekty) nad nakloněnou rovinou je poté dopočítáno ohraničení tyrkysové barvy. Uložené pozice rohů jsou vypsány v horní části záložky, výsledek zpracování obrazu je zobrazen v její spodní části pod názvem *Morfologická maska*.

Další možností je *Horizontální kalibrace roviny*, která se provádí při regulaci kuličky do fixního bohu v době jejího ustálení. Aktuální úhly náklonu roviny



Obr. 4.38 Obsah záložky *Kalibrace*

se uloží jako nulové a provede se úhlový posuv mezi skutečným úhlem vyčítaným ze servozesilovače. Při mechanické kompletaci není možné na přechodu převodovka/motor nastavit nulový úhel, také u převozu či mikro-prokluzech při řízení s vysokou dynamikou se může kumulovat úhlová chyba. Touto kalibrační procedurou ji lze jednoduše odstranit.

Poslední volbou kalibrace je *Kalibrace velikosti kuličky*. Jedná se o určení velikosti kuličky a následně její hmotnosti pomocí obrazové informace z kamery (nalezený poloměr ohraničení). Při použití této metody je však nutné brát zřetel na to, aby nebyla při jednotlivých krocích zpracování obrazu změněna velikost kuličky (např. přílišným ořezáním při prahování či filtraci).

Nachází se zde také tlačítko pro uložení aktuálního nastavení celé aplikace (*Uložit nastavení*) a tlačítko pro načtení uloženého nastavení, které se provádí při každém startu aplikace (*Načíst nastavení*).

4.6 Zpracování obrazu

V předchozí podkapitole byly obecně a z uživatelského hlediska popsány a předvedeny jednotlivé kroky práce s obrazem. V této podkapitole bude vysvětleno, jaké metody byly k dosažení prezentovaných výsledků použity a na jakých principech se zakládají. Nebude zde detailně rozebíraný zdrojový kód, spíše půjde o popis posloupnosti použitých/naprogramovaných metod a odůvodnění, proč byly použity.

Úkolem zpracování obrazu je v tomto případě vybrat jeden objekt zájmu a určit jeho 2D pozici na nakloněné rovině. Existují různé metody sledování objektu, pro účely této práce však plně postačuje jednoduchá, ale rychlá a robustní detekce sledovaného objektu na základě informace o jeho barevném odstínu - nakloněná rovina, tedy pozadí snímané scény, je barevně stálé. Robustnost a úspěšnost této metody, jako všech metod pracujících s viditelnou částí spektra, závisí velkou měrou na světelných podmínkách snímané scény - ty jsou částečně zajištěny přísvitem. Algoritmy zpracování obrazu jsou obecně náročné na výpočetní výkon, proto je vždy nutné volit při požadavku na použití v reálném čase určitý kompromis mezi rychlostí a kvalitou zpracovaného obrazu. V běžných aplikacích stále postačuje rozlišení 640 x 480 pixelů (a odpovídající varianty s odlišným poměrem stran). Při použití základních prahovacích metod, filtrace šumu a výpočtu plošného těžiště postačuje pro zpracování obrazu v reálném čase i běžně dostupný počítač s obyčejnou web-kamerou.

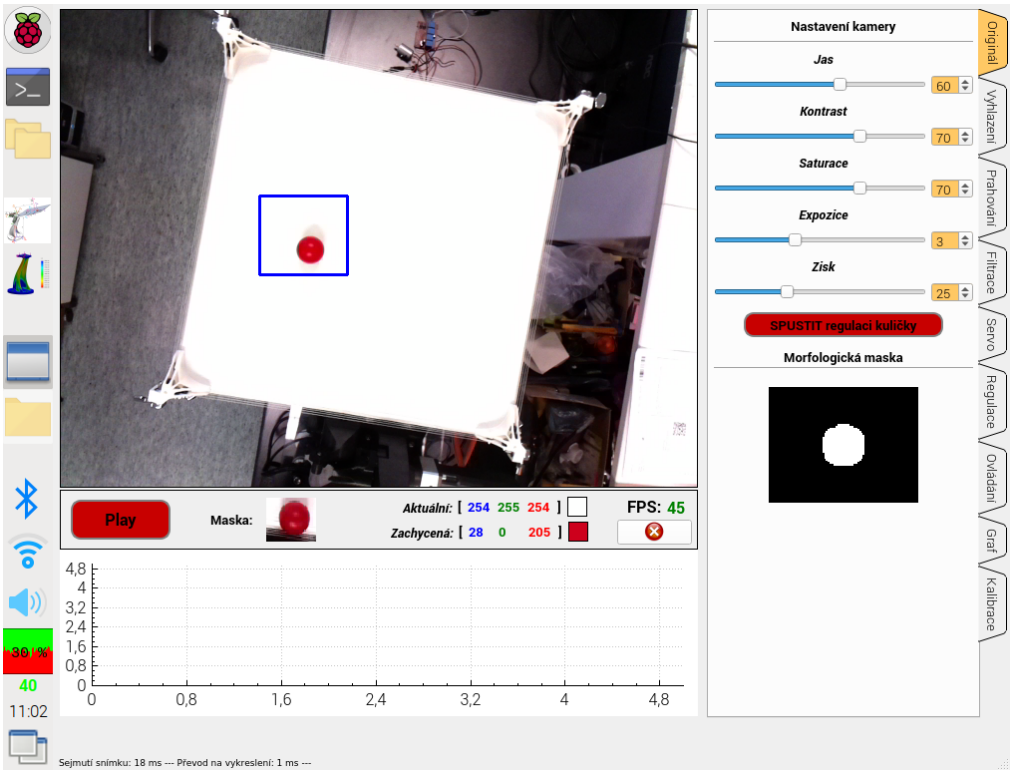
Na druhé straně existují pokročilé algoritmy sledování objektu využívající neuronové sítě spolu s histogramovou maskou sledovaného objektu, které jsou univerzálnější (s ohledem na charakter sledovaného objektu) a využívají k detekci celou matici pixelů. Tyto pokročilé metody jsou však zpravidla výpočetně mnohem náročnější a při použití v reálném čase je nutné mít odpovídající hardware.

Většina (jak základních, tak pokročilých) operací pro manipulaci s obrazem je implementována v otevřené multiplatformní knihovně OpenCV. Tato knihovna obsahuje také funkce pro parametrizaci připojené kamery (pokud to software kamery dovoluje), vyčítání obrazu, jeho následné vykreslení apod. Její funkce jsou optimalizované, neustále ve vývoji a je hojně využívána i v komerční sféře.

Veškerý kód týkající se práce s obrazem je implementován ve třídě *CameraWorker*, která běží ve vlastním paralelním vláknu tak, aby byla co možná nejméně narušována plynulost vykonávání jejich funkcí.

4.6.1 Sejmutí snímku a výběr oblasti zájmu

Podle volby počítače se vyšle do kamery požadavek na sejmutí snímku (RPi pro kameru na CSI sběrnici, PC pro kameru připojenou přes USB).



Obr. 4.39 Ukázka výběru oblasti zájmu

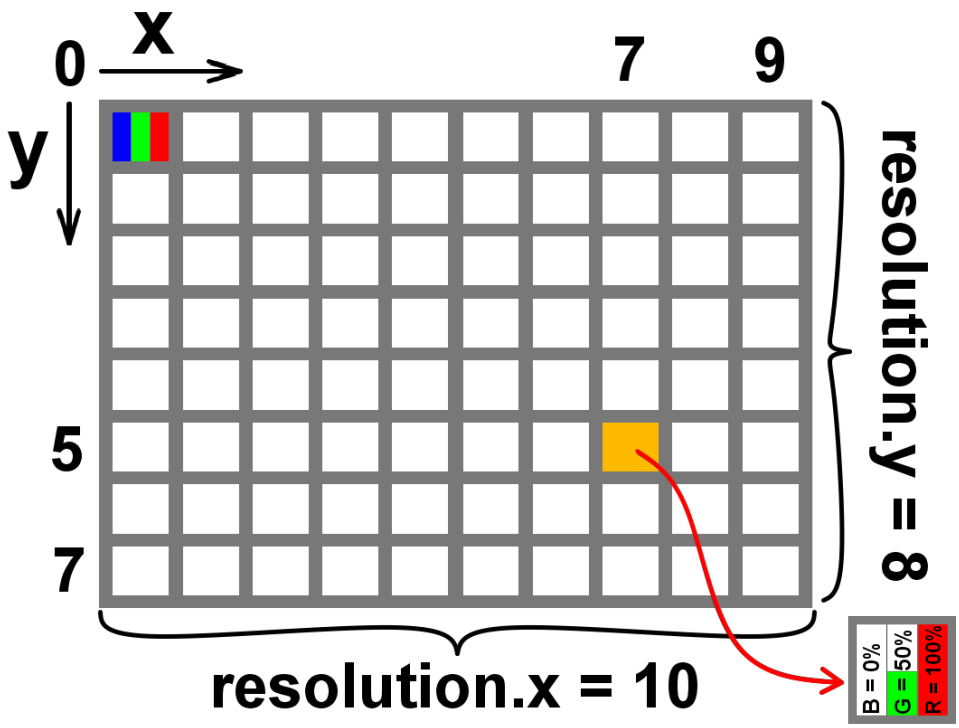
Nastavení parametrů kamery v záložce *Originál* je určeno pro RPi, protože na tento řídicí počítač je celá aplikace psána. V případě vývoje na jiném počítači je dosti pravděpodobné, že připojená kamera nemá takové možnosti nastavení, nebo mají parametry jinou škálu. Po odpovědi kamery se snímek uloží do matice určené pro zobrazení, nativně v paletě BGR a s rozlišením 640x480 pixelů. Dále

se ze snímku vyřeže vybraná oblast zájmu a uloží do matice určené ke zpracování obrazu - *desireFrame*. Výběr je realizován označením oblasti myší při stisknutém pravém tlačítku. Barva rámečku při výběru je volena podle určené barvy sledovaného objektu (buď modrá, nebo červená). Při označování je rámeček modrý, pokud má barva sledovaného objektu více červené složky než modré, nebo je červený, pokud má barva sledovaného objektu více modré složky než červené. Určení barvy ohraničení oblasti zájmu při sledování má opačnou logiku, tím je zajištěna dobrá vizuální orientace uživatele.
Veškerá práce s obrazem je dále omezena pouze na oblast zájmu. **Mimo označenou obdélníkovou zónu tedy není snímek vůbec zpracováván**, což je klíčový krok pro optimalizaci výpočetního výkonu, především s ohledem na použitý hardware (Raspberry Pi). Při označování oblasti zájmu je snímání z kamery pozastaveno a přestanou se vykreslovat všechny pomocné prvky, aby nerušily při výběru - viz Obr. 4.39. Po uvolnění pravého tlačítka myši se snímání obrazu z kamery opět spustí s již označenou oblastí zájmu, na kterou se později aplikují algoritmy zpracování obrazu.

4.6.2 Označení barvy sledovaného objektu

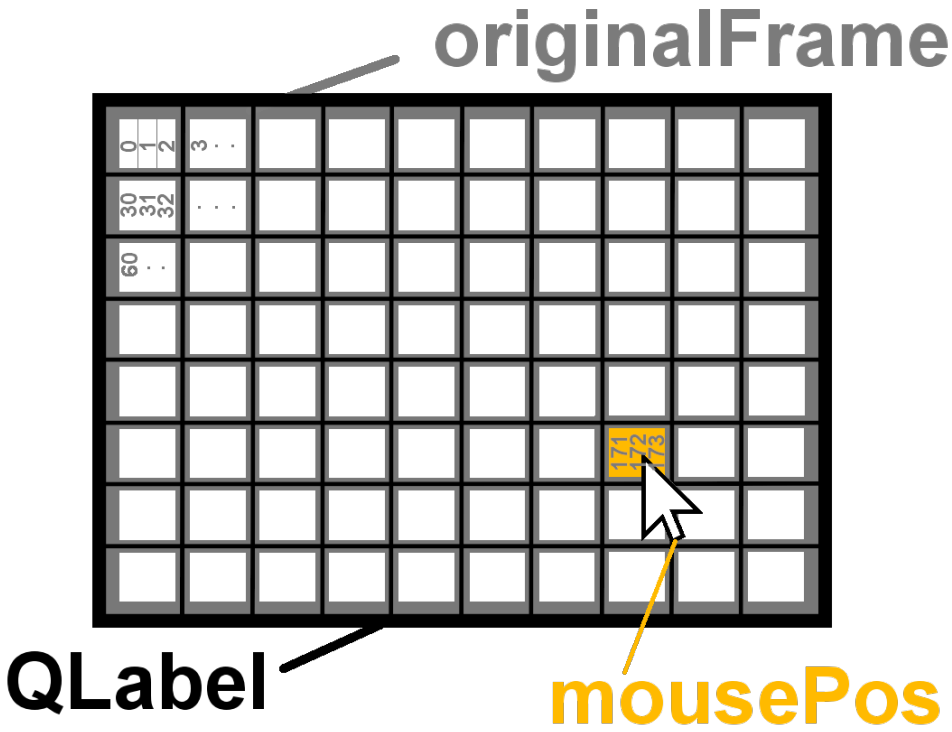
Nejpohodlnější způsob zadání barvy sledovaného objektu se jeví výběr pomocí kurzoru myši. Byla tedy implementována metoda, která ukládá žádanou barvu na pozici kurzoru myši nad vykreslovaným snímkem po dvojkliku levým tlačítkem myši. Barva se zjišťuje z originálního snímku v plné velikosti (tedy ještě před výřezem oblasti zájmu), nebo z ořezaného snímku po vyhlazení (poslední fáze, kdy je ještě snímek barevný). Její odstín se zároveň i vykreslí s výpisem úrovní jednotlivých složek do oblasti pod snímkem (pojmenováno jako *Zachycená*). Při pohybu kurzoru nad snímkem se v reálném čase také vypisuje i vykresluje barva na aktuální pozici kurzoru s periodou 30 ms. Toto omezení je zde z důvodu optimalizace výpočetního výkonu, výpis má pouze informativní charakter a překreslování je v tomto nastavení dostatečně plynulé. Bez omezení a při rychlém pohybu myši přes okno se snímkem, dosahovala vzorkovací frekvence překreslování i 1 kHz, což je okem člověka nepostřehnutelné a zbytečně výpočetně náročné.

Tuto metodu je možné využít i v průběhu regulace ve všech částech zpracování obrazu a ve všech záložkách, což vyžadovalo notné úpravy a ošetření. Patří mezi ně např. vyřešení výlučného přístupu k proměnným, které se přepisují z jiného vlákna, nebo zajištění přepočtu mezi pozicí kurzoru v ořezaném snímku roztaženém na plnou velikost okna a snímkem původní velikosti.



Obr. 4.40 Zjednodušený popis snímku z kamery

Na Obr. 4.40 je naskicován zjednodušený snímek z kamery, na které bylo nastaveno rozlišení 10 x 8 pixelů. Na 8. pozici v ose x indexované číslem 7 a 6. pozici v ose y indexované číslem 5 byl kamerou detekovaný pixel oranžové barvy, který je reprezentován podílem jednotlivých barevných složek BGR palety jako {0, 128, 256} - v případě 8 bitové barevné hloubky na kanál. Názvy proměnných jsou použity totožné jako v programu, aby se čtenář lépe zorientoval v následných ukázkách kódu - části zdrojové kódu jsou přiloženy pouze u stěžejních částí programu.



Obr. 4.41 Promítnutí originálního snímku do zobrazovacího okna a naznačení reprezentace dat

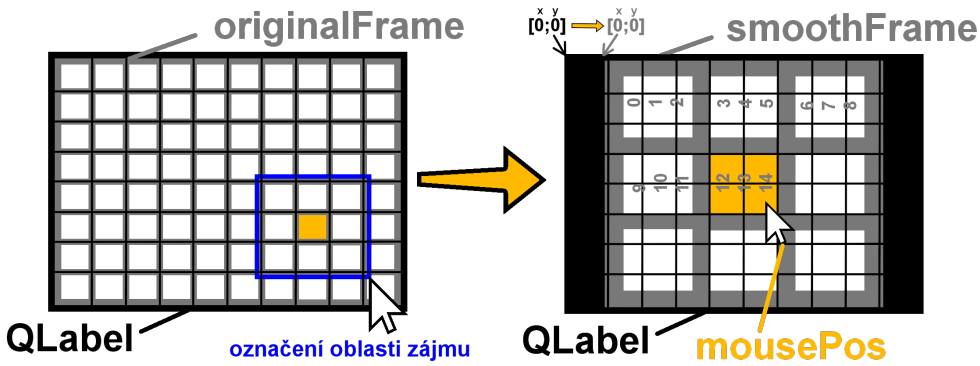
Snímek z kamery je uložen do proměnné *originalFrame* datového typu reprezentovaného jako lineární pole viz Obr. 4.41. Toto je nejjednodušší případ pro výběr barvy v neořezaném snímku, u kterého je už ale potřeba ošetřit zvolené rozlišení kamery a velikost zobrazovacího okna *QLabel*. Při rozdílné velikosti se snímek z kamery přizpůsobí velikosti zobrazovacího okna a pozice kurzoru v okně tedy nebude odpovídat pozici ve snímku v proměnné *originalFrame*, která má pevnou velikost podle zvoleného rozlišení snímku z kamery.

Při označení oblasti zájmu pouze jako části originálního snímku (což je zpravidla vždy), se do zpracování obrazu přesouvá pouze tato označená část. V záložkách pro zpracování obrazu (*Vyhlazení, Prahování, Filtrace*) je tedy v zobrazovacím okně vykreslena pouze označená část snímku, která je roztažená se zamčeným poměrem stran do celého okna. Tím je snímek přiblížen, v jednom

rozměru (podle tvaru oblasti zájmu) je roztažen na velikost zobrazovacího okna, v druhém rozměru je vycentrován a zbylý prostor je vyplněn černou barvou viz Obr. 4.42.

```
Vec3b CameraWorker::GetActualBGR(QPoint pos)
{
    ⋮
    // prepocet s ohledem na velikost vykreslovaci oblasti vuci
        rozliseni snimku
    mouseEventTmp1i = floor(mousePos.y()*resolution.x/qLableRes.x
        )*resolution.x*originalFrame.channels() + floor(mousePos.x
        ()*resolution.y/qLableRes.y)*originalFrame.channels();

    // vraceni aktualni BGR barvy bodu ve snimku, kde se nachazi
        kurzor mysi
    return Vec3b(originalFrame.data[mouseEventTmp1i],
        originalFrame.data[mouseEventTmp1i + 1], originalFrame.
        data[mouseEventTmp1i + 2]);
    ⋮
}
```

Zdr. 1 Ukázka kódu pro zachycení barvy v originálním snímku



Obr. 4.42 Proces označení oblasti zájmu a přizpůsobení snímku vykreslovacímu oknu

To přináší komplikace při zaměření barvy ve výše zmíněných záložkách, protože velikost ani pozice označené části snímku už neodpovídá snímku uloženém v

matici pro zpracování obrazu. Na tuto matici (*smoothFrame*) jsou aplikovány i funkce pro vyhlazení.

```
Vec3b CameraWorker::GetActualBGR(QPoint pos)
{
    ⋮
    else {
        // zjisteni prevodoveho pomeru vyrezane oblasti ze snimku
        mouseEventTmpf = ((float)smoothFrame.rows/resolution.y < (
            float)smoothFrame.cols/resolution.x)*(float)smoothFrame
            .cols/resolution.x + ((float)smoothFrame.rows/
            resolution.y >= (float)smoothFrame.cols/resolution.x)*(
            float)smoothFrame.rows/resolution.y;

        // kvuli vycentrovani obrazu je potreba prepocitat
            souradnice mysi
        mouseEventTmpv2i = Vec2i(floor(mousePos.x()-(qLableRes.x-
            smoothFrame.cols/mouseEventTmpf)/2), floor(mousePos.y()
            -(qLableRes.y-smoothFrame.rows/mouseEventTmpf)/2));

        // osetreni proti pristupu mimo pole (pouze nad vyrezanou
            oblasti)
        if(mouseEventTmpv2i.val[0] >= 0 && mouseEventTmpv2i.val[0]
             < smoothFrame.cols/mouseEventTmpf && mouseEventTmpv2i.
            val[1] >= 0 && mouseEventTmpv2i.val[1] < smoothFrame.
            rows/mouseEventTmpf) {

            // prepocet s ohledem na velikost vykreslovaci oblasti
            mouseEventTmp1i = floor(mouseEventTmpv2i.val[1]*
                resolution.x/qLableRes.x*mouseEventTmpf)*
                smoothFrame.cols*smoothFrame.channels() + floor(
                mouseEventTmpv2i.val[0]*resolution.y/qLableRes.y*
                mouseEventTmpf)*smoothFrame.channels();

            return Vec3b(smoothFrame.data[mouseEventTmp1i],
                smoothFrame.data[mouseEventTmp1i + 1], smoothFrame.
                data[mouseEventTmp1i + 2]);
        }
    }
}
```

Zdr. 2 Ukázka kódu pro zachycení barvy v ořezaném snímku

Upřesnění výběru sledované barvy se ale vždy provádí na ořezaném snímku, protože je jednak vyhlazený a také je přiblížený, takže je zaměření přesnější. Pro přepočet mezi pozicí kurzoru myši v *QLabel* a mezi pozicí v *smoothFrame* byl tedy navržen výše uvedený přístup.

Nejdříve je zjištěný převodový poměr, s kterým se snímek roztáhne na velikost *QLabel*. Vybere se převodový poměr toho rozměru (x nebo y), který se roztáhne do plné velikosti *QLabel*, protože jedině ten odpovídá definici. Kvůli vycentrování ořezaného snímku při zobrazení se dále provede přepočet nulových souřadnic kurzoru myši do levého horního rohu *smoothFrame* viz Obr. 4.42. Tím se provede poziční posun v té ose snímku, která není roztažena na plnou velikost okna *QLabel*. Dále se po ověření, že se kurzor myši nachází nad snímkem, vypočítá skutečná pozice v ořezaném snímku. Zde už je logika stejná jako v předchozím případě pro originální snímek. Jak je vidět v přiložených ukázkách zdrojového kódu, program je detailně komentován. Skutečný snímek z kamery odpovídající této části je zobrazený např. na Obr. 4.39 - výběr oblasti zájmu v hlavním okně, přizpůsobená oblast zájmu (po obrazovém zpracování) pod nadpisem *Morfologická maska*.

4.6.3 Vyhlazení ořezaného snímku

Po výběru oblasti zájmu, jejím výřezu z originálního snímku a označení barvy objektu, na který se mají aplikovat algoritmy zpracování obrazu, se provede první krok - vyhlazení/rozmazání snímku. Jak již bylo dříve zmíněno, je v programu použito celkem 9 parametrizovatelných filtrů. Za nejlepší kompromis v této práci je považováno *Gaussiánské rozmazání* o velikosti kernelu 2. Ostatní filtry buď nepřinášejí tak dobré výsledky, nebo jsou výpočetně náročné. Všechny použité filtry jsou kombinace optimalizovaných funkcí knihovny OpenCv [50], a jsou to:

Žádný
Pokud není potřeba snímek vyhladit/rozmazat, lze tuto operaci vynechat. V takovém případě se ale musí předat kopie vstupní matice.

```
// bez rozmazani
case 0:
    // se snimkem se nic nedela, takze se preda original
    smoothFrame = desiredFrame.clone();
    break;
```

Zdr. 3 Příklad vynechání kroku vyhlazení snímku

Filter2D

Tato funkce *filter2D* je obecný 2D filtr, který provede konvoluci mezi vstupním snímkem a kernelem (konvoluční maticí). Přesněji řečeno, podle zápisu tato funkce provede spíše korelaci, protože kernel není zrcadlený podél horizontální osy. Střed konvoluční matice, její velikost i tvar, extrapolace oblasti kernelu u krajních pixelů, barevná hloubka atd. je plně na programátorovi. V tomto případě je nastaven střed kernelu nad pixelem, s kterým se provádí konvoluce a váhové koeficienty konvoluční matice jsou rozprostřeny rovnoměrně, tedy váha pixelů ve středu je stejná, jako váha pixelů u okraje. Výpočetní náročnost této metody lze zařadit mezi vyšší (má velké množství nastavitelných parametrů) a rapidně roste s velikostí kernelu. Velikost kernelu musí být liché číslo větší než 1 a při velikosti > 9 se již výpočetní nároky nijak zásadně nezvyšují - používá se diskrétní Fourierova transformace (DFT).

```
// Filter2D
case 1:
    tmp1i = smoothKernelSize*2 + 1;  // velikost kernelu musi byt
        liche cislo vetsi nez 1
    smoothKernel = Mat::ones(tmp1i, tmp1i, CV_32F)/(float)(tmp1i*
        tmp1i);    // format kernelu pro "filter2D"
    filter2D(desiredFrame, smoothFrame, -1, smoothKernel); //
        korelace snimku s kernelem
    break;
```

Zdr. 4 Příklad použité funkce *filter2D*

Homogenní rozmazání

Je použita funkce *blur*, která rozostří obraz pomocí normalizovaného „box“ filtru. Tento filtr je nejjednodušší a výpočetně nejméně náročný. Každý výstupní pixel je průměrem pixelů v jeho okolí, kdy všechny okolní pixely přispívají stejnou vahou. Toto okolí je definováno velikostí kernelu (nemusí být čtvercový) -

nastavuje se šířka a výška konvoluční matice. Další parametry k nastavení jsou pozice konvoluční matice nad aktuálním pixelem a typ extrapolace kernelu u okrajů matice.

```
// Homogenni rozmazani
case 2:
    blur(desiredFrame, smoothFrame, Size(smoothKernelSize,
        smoothKernelSize));
    break;
```

Zdr. 5 Příklad použité funkce *blur*

Gaussiánské rozmazání

Ve funkci *GaussianBlur* se oproti předchozí funkci *blur* nepočítá se stejnými váhovými koeficienty (v kernelu), ale s kernelem s váhovými koeficienty s Gaussovým rozložením. Už z definice vyplývá, že velikost kernelu musí být kladná a lichá - kvůli symetričnosti Gaussovy funkce. Velikost kernelu je nastavitelná, stejně jako směrodatná odchylka v obou osách a typ extrapolace. Z důvodu tvaru kernelu není možné tuto funkci optimalizovat tak, jako *blur* a má nízké až střední výpočetní nároky. Je velice účinná především v odstranění Gaussova šumu.

```
// Gaussianske rozmazani
case 3:
    tmp1i = smoothKernelSize*2 + 1;
    GaussianBlur(desiredFrame, smoothFrame, Size(tmp1i, tmp1i), 0,
         0);
    break;
```

Zdr. 6 Příklad použité funkce *GaussianBlur*

Střední rozmazání

Až doteď byly popisovány filtry, které obraz opravdu rozmazaly, místo toho aby jej vyhladily, jak indikuje název záložky. Funkce *medianBlur* už obraz dá se říci částečně vyhladí se zachováním původních barevných odstínů, je to ovšem vykoupeno výpočetními nároky tohoto filtru. Funkce spočítá medián všech pixelů pod kernelem a středový pixel nahradí touto střední hodnotou. Tento filtr je velice účinný při odstranění tzv. impulzního šumu, nazývaného také jako „salt-and-pepper noise“. Předchozí filtry nahradily středový pixel hodnotou, která ani nemusela být v původním snímku. To však není případ filtru *medianBlur*, který

vždy nahradí středový pixel skutečnou hodnotou nějakého pixelu v původním snímku. To snižuje výsledný šum a zachovává původní barevný odstín výsledného snímku. Velikost kernelu musí být opět kladné liché číslo a je to jediný nastavitelný parametr. Výpočetní náročnost tohoto typu filtru je vysoká.

```
// Stredni rozmazani
case 4:
    tmp1i = smoothKernelSize*2 + 1;
    medianBlur(desiredFrame, smoothFrame, tmp1i);
    break;
```

Zdr. 7 Příklad použité funkce *medianBlur*

Bilaterální rozmazání

Všechny předchozí zmíněné filtry původní obraz více či méně rozmazaly a vždy měly tendenci rozmazávat hrany. Funkce *bilateralFilter* je první filtr, který odstraňuje šum a zároveň zachovává ostré barevné přechody. Je variantou Gaussova filtru, který ovšem nepočítá pouze vážený průměr okolních pixelů, ale zohledňuje i jejich intenzitu v závislosti na intenzitě středového pixelu - je tedy nejen funkcí prostoru, ale i intenzity. Jedna Gaussova složka zajistí, že prostorově vzdálené pixely budou mít menší vliv na celkový výsledek, druhá Gaussova složka zase zaručí, že pouze ty pixely, které mají blízkou intenzitu (barvu) intenzitě středového pixelu, budou zahrnuty do výpočtu s váhou ovlivňující výsledek. Tato metoda dosahuje viditelných výsledků až při větších velikostech kernelu, což je ale z pohledu výpočetní náročnosti neúnosné. Má perfektní výsledky, ale velmi vysoké nároky na výpočetní výkon. Mezi nastavitelné parametry patří velikost kernelu, typ extrapolace a především vliv jednotlivých Gaussových složek (pozice i barevná intenzita).

```
// Bilateralni rozmazani
case 5:
    bilateralFilter(desiredFrame, smoothFrame, smoothKernelSize,
        smoothKernelSize*2, smoothKernelSize/2);
    break;
```

Zdr. 8 Příklad použité funkce *bilateralFilter*

Morfologie - Otevření

Dále jsou v aplikaci k dispozici 4 typy různých kombinací morfologických funkcí.

Jejich uplatnění je spíše u filtrace obrazu, kde jsou také základní morfologické operace vysvětleny a používají se na binární obraz. Pro účely vyhlazení barevného snímku o plné velikosti jsou výpočetně středně náročné a tyto nároky rapidně rostou při zvětšování strukturního elementu (jádra/kernelu).
První z nich je *Morfologie - Otevření*. Tento typ operace se používá především k odstranění zbylého šumu, nebo menších objektů - zde obstarává vyhlazení obrazu. *Otevření* je složeno z 2 po sobě jdoucích základních morfologických operací, kterými jsou **eroze** a **dilatace** - popsány v části *Filtrace*.
Nejprve je na snímek aplikována eroze, poté dilatace. Výsledkem je potlačení (nebo lépe upozadění) malých barevně odlišných oblastí, a poté „slití“ výsledného obrazu pomocí dilatace. Používá se při vyhlazení snímku, který je poznamenaný šumem vně/okolo objektu zájmu.

```
1 // Morfologie - otevreni
2 case 6:
3     // vytvoreni elementu pro morfologicke otevreni, skladajici se
          z velikosti a tvaru kernelu
4     element = getStructuringElement(smoothShape, Size(2 *
          smoothKernelSize + 1, 2 * smoothKernelSize + 1), Point(
          smoothKernelSize, smoothKernelSize));
5     morphologyEx(desiredFrame, smoothFrame, MORPH_OPEN, element);
6     break;
```

Zdr. 9 Příklad použité funkce *morphologyEx* typu *OPEN*

Morfologie - Uzavření

Další morfologickou operací je *Uzavření*, tedy dilatace následovaná erozí. Oproti předchozímu *Otevření* se v obrazu nejprve sjednotí blízké, malé, barevně odlišné oblasti a až poté se provede eliminace malých osamostatněných objektů, které se po dilataci „neslily“ do jednoho většího celku. Oblast použití je pro vyhlazení/eliminaci šumu, který se vyskytuje uvnitř objektu zájmu. Stejně jako v předchozím případě je u morfologických operací k dispozici nejen volba velikosti jádra (strukturního elementu), ale i jeho tvaru (čtverec, kříž, elipsa) - seřazeno od nejméně po nejvíce výpočetně náročné.

```
// Morfologie - uzavreni
case 7:
    // vytvoreni elementu pro morfologicke uzavreni, skladajici se
        z velikosti a tvaru kernelu
    element = getStructuringElement(smoothShape, Size(2 *
        smoothKernelSize + 1, 2 * smoothKernelSize + 1), Point(
        smoothKernelSize, smoothKernelSize));
    morphologyEx(desiredFrame, smoothFrame, MORPH_CLOSE, element);
    break;
```

Zdr. 10 Příklad použité funkce *morphologyEx* typu *CLOSE*

Otevření + Uzavření

Morfologické *Otevření* následované *Uzavřením* kombinuje obě dříve uvedené operace. Výpočetní čas se pochopitelně oproti samostatnému *Otevření* a *Uzavření* opět prodlužuje a tuto kombinaci lze použít v případě, že je potřeba nejdříve eliminovat šum okolo objektu zájmu, poté sjednotit výsledný obraz dvakrát po sobě následující dilatací a následně znovu provést potlačení zbylých malých barevně odlišných částí (eroze + dilatace + dilatace + eroze). Výsledkem je prohloubené vyhlazení obrazu a dosahuje lepších výsledků, než samotné *Otevření*.

```
// Otevreni + uzavreni
case 8:
    element = getStructuringElement(smoothShape, Size(2 *
        smoothKernelSize + 1, 2 * smoothKernelSize + 1), Point(
        smoothKernelSize, smoothKernelSize));
    morphologyEx(desiredFrame, smoothFrame, MORPH_OPEN, element);
    morphologyEx(smoothFrame, smoothFrame, MORPH_CLOSE, element);
    break;
```

Zdr. 11 Příklad použité funkce *morphologyEx* typu *OPEN* + *CLOSE*

Uzavření + Otevření

Morfologické *Uzavření* následované *Otevřením* je pouze obrácená předchozí posloupnost operací, tedy dilatace následovaná dvěma erozemi a poté opět dilatací. Výsledkem je mírně prohloubené vyhlazení obrazu oproti *Uzavření*, s eliminací zbylých šumů.

```
// Uzavreni + otevreni
case 9:
    element = getStructuringElement(smoothShape, Size(2 *
        smoothKernelSize + 1, 2 * smoothKernelSize + 1), Point(
        smoothKernelSize, smoothKernelSize));
    morphologyEx(desiredFrame, smoothFrame, MORPH_CLOSE, element);
    morphologyEx(smoothFrame, smoothFrame, MORPH_OPEN, element);
    break;
```

Zdr. 12 Příklad použité funkce *morphologyEx* typu *CLOSE + OPEN*

4.6.4 Prahování vyhlazeného snímku

Po vyhlazení je na snímek aplikováno prahování, které je popsané v odstavci *Záložka Prahování*. Po výběru vhodné barevné palety (BGR, LAB nebo HSV) podle aktuální situace se převede označená barva do zvolené barevné palety a provede se ošetření tolerancí pro prahování.

```
switch(thresholdType) {
// BGR paleta
case 1:
    // original uz je v BGR, takze neni treba zadny prevod
    thresholdCenterColor = thresholdCenterBGR;
    break;

// HSV paleta
case 2:
    // prevedeni odchyceneho bodu (dvojklik mysi) pro prahovani do
        palety HSV
    cvtColor(thresholdCenterMat, thresholdCenterMat, COLOR_BGR2HSV
        );

    // ulozeni tohoto noveho stredu pro prahovani v HSV palete
    thresholdCenterColor = thresholdCenterMat.at<Vec3b>(0,0);
    break;

// LAB paleta
case 3:
    // prevedeni do palety LAB
```

```
20      cvtColor(thresholdCenterMat, thresholdCenterMat, COLOR_BGR2Lab
            );
21      // ulozeni stredu v LAB palete
22      thresholdCenterColor = thresholdCenterMat.at<Vec3b>(0,0);
23      break;
24  }
```

Zdr. 13 Převod označené barvy do zvolené barevné palety

```
1   if((thresholdCenterColor.val[0] - thresholdMin.val[0]) < 0)
2       tmp1v3b.val[0] = 0;
3   else
4       tmp1v3b.val[0] = thresholdCenterColor.val[0] - thresholdMin.
            val[0];
5
6   if((thresholdCenterColor.val[0] + thresholdMax.val[0]) > 255)
7       tmp2v3b.val[0] = 255;
8   else
9       tmp2v3b.val[0] = thresholdCenterColor.val[0] + thresholdMax.
            val[0];
```

Zdr. 14 Ukázka ošetření rozsahů pro první barevnou složku

Poté se do zvolené barevné palety převede i celý snímek z kamery (nebo přesněji řečeno vybraná oblast zájmu, na kterou už bylo aplikováno vyhlazení). Zatím byl do požadované barevné palety převeden pouze vektor odchycené barvy.

```
1   switch(thresholdType) {
2   case 1:
3       // "smoothFrame" uz je v BGR formatu, takze nemusim nic
            prevadet. Minimum a maximum je pro kazdou barvu, takze
            jejich velikost je 3
4       inRange(smoothFrame, Scalar(tmp1v3b.val[0], tmp1v3b.val[1],
            tmp1v3b.val[2]), Scalar(tmp2v3b.val[0], tmp2v3b.val[1],
            tmp2v3b.val[2]), thresholdFrame);
5       break;
6
7   case 2:
8       // prevedeni do palety HSV a oprahovani
9       cvtColor(smoothFrame, tmpMat3b, COLOR_BGR2HSV);
10      inRange(tmpMat3b, Scalar(tmp1v3b.val[0], tmp1v3b.val[1],
            tmp1v3b.val[2]), Scalar(tmp2v3b.val[0], tmp2v3b.val[1],
```

```
        tmp2v3b.val[2]), thresholdFrame);
    break;

case 3:
    // prevedeni do palety LAB a oprahovani
    cvtColor(smoothFrame, tmpMat3b, COLOR_BGR2Lab);
    inRange(tmpMat3b, Scalar(tmp1v3b.val[0], tmp1v3b.val[1],
        tmp1v3b.val[2]), Scalar(tmp2v3b.val[0], tmp2v3b.val[1],
        tmp2v3b.val[2]), thresholdFrame);
    break;
}
```

Zdr. 15 Převedení vyhlazeného snímku do zvolené barevné palety a jeho oprahování

Výsledkem prahování bude černobílý snímek o velikosti oblasti zájmu, ve kterém bílá část reprezentuje ponechané objekty a černá prázdné pozadí, který bylo oprahováním odstraněno. Ve snímku se může stále vyskytovat šum, objektů může být více, mohou být neúplné, rozdělené apod. Proto je na řadě filtrace snímku.

4.6.5 Filtrace oprahovaného snímku

Základem operace nazvané jako filtrace jsou již dříve zmíněné morfologické funkce. Jsou jimi *Otevření* a *Uzavření*, které byly už rozebrány v odstavci popisující operace *vyhlazení snímku*. Nyní jsou již aplikovány na binární (černobílou) obrazovou matici, proto půjde nastínit, na jakém principu pracují jejich základní dva morfologické operátory - eroze a dilatace. Zdrojový kód je obdobný jako v případě vyhlazení, není tedy potřeba žádná ukázka implementace.

Eroze:

Název této operace mnohé napovídá (princip je stejný, jako u eroze půdy) - okraje všech zbylých objektů po oprahování budou odstraněny a ty se tedy zmenší o velikost jádra. Strukturní element se totiž postupně posunuje obrazem a zjišťuje, jestli jsou všechny pixely pod ním bílé. Pokud to tak není, celá oblast pod jádrem se překryje tvarem jádra (nastaví na černou barvu). Tím se postupně zmenšuje bílá oblast od okrajů objektů dovnitř. Aby se střed prvků neposunoval, je velikost jádra vždy liché číslo. Po pouhé erozi by sice došlo k odstranění malých

částic (šumu) z obrazu, ale zároveň by se tak ořezaly všechny objekty ve snímku. Pro zachování původní velikosti se na snímek následně aplikuje i dilatace o stejné velikosti i se stejným tvarem jádra.

Dilatace:
Je opakem eroze, jádro se opět posunuje obrazem a pokud se v oblasti pod jádrem objeví alespoň jeden bílý pixel, celá oblast ve snímku se překryje tvarem jádra (nastaví se na bílou barvu). Tím se rozrůstají hranice objektů, sjednocují blízké objekty a zacelují díry. Také se obnovuje původní velikost prvků po erozi, proto se zde nepoužívá samotná eroze ani dilatace, vždy jsou v páru a navazují na sebe (podle jejich pořadí se tento pár nazývá *Otevření* nebo *Uzavření*).
V aplikaci je možné měnit pořadí, nebo morfologické operace vyřadit úplně, ale zpravidla vždy je aplikováno nejdříve *Uzavření*, poté *Otevření*. Velikost jádra pro obě funkce nemusí být stejná (to platí pouze pro po sobě jdoucí operátory eroze a dilatace), tvar je volen z důvodu minimalizace výpočetních nároků čtvercový (kříž nepřinášejí lepší výsledky, elipsa v případě malé velikosti jádra pouze zanedbatelně, ale má zhruba 2× vyšší výpočetní nároky).

4.6.6 Detekce hran vyfiltrovaných objektů

Po filtraci může zůstat ve snímku stále více objektů, nebo se může v průběhu zpracování obrazu další objekt objevit. S tím je nutné počítat, a proto je potřeba zavézt strategii, která rozhodne, na jaký objekt se aplikace zaměří a co se stane s objekty dalšími. Byla vybrána metoda detekce hran, která nalezne všechny osamocené objekty, vybere největší z nich, vytvoří kolem něj nejmenší možnou opsanou kružnici, její střed přepočítá do rozlišení původní velikosti snímku a uloží ho jako skutečnou pozici objektu zájmu. Objektem zájmu je koule, proto i při nedokonalé identifikaci tvaru bude díky této metodě zaměřen její střed. Pokud nebude detekovaný žádný objekt, nastaví se střed oblasti zájmu na střed nakloněné roviny a čeká se, dokud se v této oblasti neobjeví alespoň jeden úspěšně identifikovaný objekt.

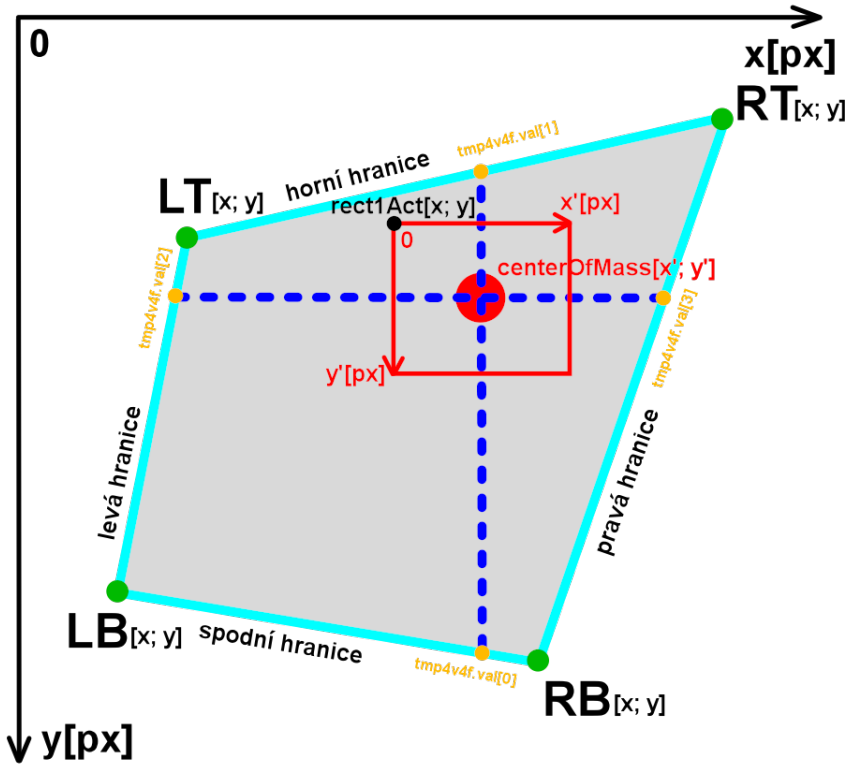
```
if(edgeDetection) {
    // nalezeni obrysu ve vyfiltrovanem obrazu
    findContours(tmpMat1b, contours, RETR_CCOMP,
        CHAIN_APPROX_SIMPLE);
    ⋮
    if(contours.size() > 0) {
       ⋮
        for(i = 0; i< contours.size(); i++)
        {
            // zjisteni velikosti obrysu
            tmp1f = contourArea(contours[i]);
            // nalezeni nejvetsiho obrysu
            if(tmp1f > tmp1i) {
                // zjisteni stredu a polomeru opsane kruznice
                    nejvetsiho objektu
                minEnclosingCircle(contours[i], centerOfMass,
                    tmp2f);
                // prepocet na cely neorezany snimek
                centerOfMass = Point2f(centerOfMass.x + rect1Act.x
                    , centerOfMass.y + rect1Act.y);

                // rovnice primky z bodu "plateLBCorner" do bodu "
                    plateRBCorner" ve smernicovem tvaru - spodni
                    hranice
                tmp4v4f.val[0] = (float)(plateRBCorner.y -
```

```
                    plateLBCorner.y) / (plateRBCorner.x -
                    plateLBCorner.x) * (centerOfMass.x -
                    plateLBCorner.x) + plateLBCorner.y;
20              // pro rovnice dalsich primek je logika stejna
21              ⋮
22              // kontrola, jestli je objekt v kalibrovanem
                    ohraniceni
23              if(centerOfMass.y < tmp4v4f.val[0] && centerOfMass
                    .y > tmp4v4f.val[1] && centerOfMass.x > tmp4v4f
                    .val[2] && centerOfMass.x < tmp4v4f.val[3]) {
24                  objectDetected = true;
25                  tmp1i = round(tmp1f);
26                  biggestContourIndex = i;
27                  radius = tmp2f;
28              }
29          }
30      }
31      // pokud byly nejake objekty detekovany, ulozim polomer a
            pozici nejvetsiho z nich
32      if(objectDetected) {
33          minEnclosingCircle(contours[biggestContourIndex],
                centerOfMass, radius);
34          centerOfMass = Point2f(centerOfMass.x + rect1Act.x,
                centerOfMass.y + rect1Act.y);
35      }
36      else {
37          // pokud nebyl zadny objekt detekovany
38          NoObjectDetected();
39      }
40  }
41  // pokud nebyl zadny objekt detekovany
42  else {
43      NoObjectDetected();
44  }
45 }
```

Zdr. 16 Výběr největšího nalezeného obrysu oprahovaného snímku v ohraničení

Součásti tohoto bloku operací je i kontrola, jestli vybraný objekt leží ve vytyčeném ohraničení nakloněné roviny - Zdr. 16, řádek 23. Výpočet ohraničení navazuje na výsledek přepočtů rohů nakloněné roviny (kapitola 4.7.3), vycházejících

z velice komplexních kinematických přepočtů uvedených v kapitole 4.7.4. Zde už se pouze proloží nalezené body v rozích roviny rovnicemi přímky ve směrnicovém tvaru, dosadí se do nich zjištěný střed sledovaného objektu (přepočítaný na celý neořezaný snímek) a zjistí se, jestli objekt stále leží uvnitř definovaných hranic - Obr. 4.43 + obecně zapsané rovnice 4.1. Pokud je objekt mezi hranicemi, uloží se, pokud ne, vyřadí se z výpočtu a hledá se nový největší objekt, který leží v ohraničení.



Obr. 4.43 Kontrola pozice kuličky mezi hranicemi roviny

$$\begin{array}{llll} y = f(x) & y = \frac{b_y - a_y}{b_x - a_x} \cdot (x - a_x) + a_y & \text{pro} & tmp4v4f.val[0,1] \\ x = f(y) & x = \frac{b_x - a_x}{b_y - a_y} \cdot (y - a_y) + a_x & \text{pro} & tmp4v4f.val[2,3] \end{array} \tag{4.1}$$

Konkrétní zápis vypadá následovně:

$$\begin{aligned}
\text{centerOfMass}_y &< \frac{RB_y - LB_y}{RB_x - LB_x} \cdot (\text{centerOfMass}_x - LB_x) + LB_y \\
\text{centerOfMass}_y &> \frac{RT_y - LT_y}{RT_x - LT_x} \cdot (\text{centerOfMass}_x - LT_x) + LT_y \\
\text{centerOfMass}_x &> \frac{LB_x - LT_x}{LB_y - LT_y} \cdot (\text{centerOfMass}_y - LT_y) + LT_x \\
\text{centerOfMass}_x &< \frac{RB_x - RT_x}{RB_y - RT_y} \cdot (\text{centerOfMass}_y - RT_y) + RT_x
\end{aligned} \tag{4.2}$$

Pokud všechny nerovnice v 4.2 platí, znamená to, že se kulička nachází stále na nakloněné rovině (přepis do kódu ve Zdr. 16, řádek 19 až 23).
V případě, že se detekce hran nemá provádět, zjišťuje se pozice objektu zájmu z celého vyfiltrovaného obrazu pomocí výpočtu plošného váhového rozložení pixelů (plošného těžiště) - Zdr. 17.

```
1  else{
2      // nalezeni plosneho rozlozeni vyfiltrovaneho obrazu
3      moment = moments(tmpMat1b, true);
4
5      // pokud se nejaka oblast nasla, najdu jeji teziste
6      if(moment.m00 != 0) {
7          // signalizace, ze byl objekt detekovany
8          objectDetected = true;
9          // ulozeni pozice (teziste) sledovaneho objektu
10         centerOfMass = Point(round(moment.m10/moment.m00) +
               rect1Act.x, round(moment.m01/moment.m00) + rect1Act.y);
11
12         // matematicke vyjadreni ohraniceni naklonene roviny a
               kontrola, jestli je objekt mezi temito hranicemi, je
               stejne jako drive
13     }
14
15     // pokud nebyl zadny objekt detekovany, nastavim stred oblasti
           zajmu a signalizuju, ze nebyl objekt detekovany
16     else {
17         NoObjectDetected();
18     }
19 }
```

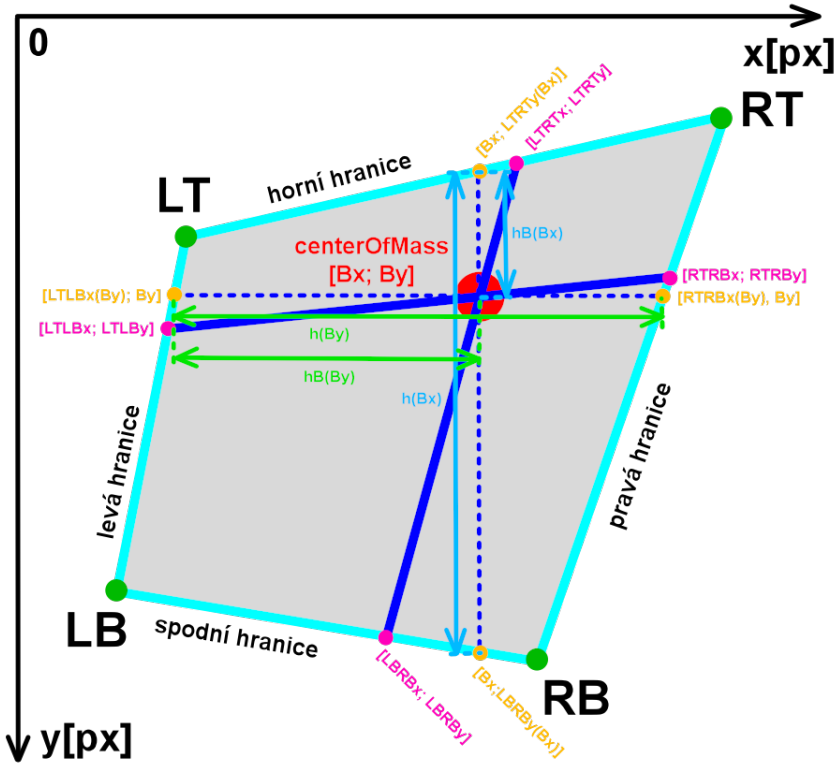
Zdr. 17 Zjištění těžiště výpočtem váhového rozložení pixelů

V takovém případě se ale musí počítat se zjevnými následky při nedokonalém

prahování + filtraci. Tento přístup je aplikován zpravidla pouze při kalibraci rohů nakloněné roviny, protože tam není žádané, ba ani možné, vyřazovat objekty nespadající do ohraničení, které se právě při kalibraci nastavuje.

4.6.7 Sledování jednoho vybraného objektu zájmu

Pokud je vyžadováno řízení polohy kuličky (objektu zájmu), ale není uvažováno využít ručního režimu ovládání libovolné smyčky pomocí gyroskopického ovladače, ani řízení podřízené smyčky (náklon roviny, její úhlová rychlost nebo proud motorů), není sledování potřeba a bude vypnuté. V takovém případě se nastaví rovina do výchozí (horizontální) polohy a pozice oblasti zájmu se nemění.
V opačném případě je nutné pro úspěšné sledování vyřešit několik stěžejních částí. Nejdříve se přenastaví pozice oblasti zájmu do středu sledovaného objektu. To zajistí, že se bude zpracovávaná oblast posouvat se změnou pozice kuličky a bude tedy vždy v ideálním případě v jejím středu. Dále se ošetřuje, aby oblast zájmu nezasahovala za ohraničení nakloněné roviny a nebyly tak zpracovávány části mimo pracovní zónu. V případě přesahu se hranice oblasti zájmu zafixují na ohraničení nakloněné roviny.
Následuje výpočet spojnic $\parallel$ s hranicemi nakloněné roviny, které protínají kuličku v jejím středu. Spojnice budou mít sklon závisející na poloze kuličky v pracovní ploše, náklonu a poměru délek obou odpovídajících hraničících linií roviny a také na pozici kuličky v obrazové matici pixelů - Obr. 4.44 + rovnice 4.3. **Je potřeba totiž zjistit reálnou pozici kuličky na nakloněné rovině, ne pouze pozici v matici pixelů (snímku)**. Aby byl přepočet univerzální, je nutné uvažovat kombinaci perspektivního zkreslení, natočení nakloněné roviny okolo optické osy kamery a také její vychýlení vůči této optické ose. Každá strana roviny má tedy (kromě jedné konfigurace - nakloněná rovina $\parallel$ s rovinou kamery - výchozí horizontální poloha roviny) různou délku a každá dvojice stran svírá jiný úhel. V konečném důsledku to znamená, že z pohledu kamery bude nakloněná rovina nepravidelným obecným čtyřúhelníkem (konvexní různoběžník), který se musí zpětnými geometrickými transformacemi narovnat do původního čtvercového tvaru, aby bylo možné určit reálnou pozici kuličky. Náčrt obecné situace je na Obr. 4.44, výpočty pro určení spojnic (souvislé modré úsečky v náčrtu) jsou v rovnici 4.3 a jejich přepis do kódu programu ve Zdr. 18.



Obr. 4.44 Nalezení spojnic kuličky ‖ s hranicemi nakloněné roviny

$$
\begin{aligned}
LBRB_y\left(B_x\right) &= \frac{RB_y - LB_y}{RB_x - LB_x} \cdot \left(\text{centerOfMass}_x - LB_x\right) + LB_y \\
LTRT_y\left(B_x\right) &= \frac{RT_y - LT_y}{RT_x - LT_x} \cdot \left(\text{centerOfMass}_x - LT_x\right) + LT_y \\
rat_y &= \frac{hB(B_x)}{h(B_x)} = \frac{centerOfMass_y - LTRT_y(B_x)}{LBRB_y(B_x) - LTRT_y(B_x)} \\
LTLB_y &= \left(LB_y - LT_y\right) \cdot rat_y + LT_y \\
LTLB_x &= \frac{LB_x - LT_x}{LB_y - LT_y} \cdot \left(LTLB_y - LT_y\right) + LT_x \\
RTRB_y &= \left(RB_y - RT_y\right) \cdot rat_y + RT_y \\
RTRB_x &= \frac{RB_x - RT_x}{RB_y - RT_y} \cdot \left(RTRB_y - RT_y\right) + RT_x
\end{aligned}
\tag{4.3}
$$

```
// prenastaveni oblasti zajmu
rect1Act.x = centerOfMass.x - floor(sizeDesiredArea.x/2);
rect2Act.x = centerOfMass.x + round(sizeDesiredArea.x/2);
rect1Act.y = centerOfMass.y - floor(sizeDesiredArea.y/2);
rect2Act.y = centerOfMass.y + round(sizeDesiredArea.y/2);
```

```
6
7  // osetreni presahu mimo zadanou oblast
8  CheckAreaBorderCornerEnable();
9
10 // vypocet spojnic kolmych k naklonene rovine a prochazejicich
       kulickou s ohledem na orientaci naklonene roviny a jeji
       zkresleni od kamery
11 // horizontalni cara ... y  = f(x)
12 tmp1f = (float)(plateRBCorner.y - plateLBCorner.y) / (
       plateRBCorner.x - plateLBCorner.x) * (centerOfMass.x -
       plateLBCorner.x) + plateLBCorner.y;    // LBRBy(Bx)
13 tmp2f = (float)(plateRTCorner.y - plateLTCorner.y) / (
       plateRTCorner.x - plateLTCorner.x) * (centerOfMass.x -
       plateLTCorner.x) + plateLTCorner.y;    // LTRTy(Bx)
14 tmp3f = (centerOfMass.y - tmp2f) / (tmp1f - tmp2f); // rat_y
15
16 tmp1v4f.val[0] = (plateLBCorner.y - plateLTCorner.y) * tmp3f +
       plateLTCorner.y;   // LTLBy
17 tmp1v4f.val[1] = (float)(plateLBCorner.x - plateLTCorner.x) / (
       plateLBCorner.y - plateLTCorner.y) * (tmp1v4f.val[0] -
       plateLTCorner.y) + plateLTCorner.x;   // LTLBx
18
19 tmp1v4f.val[2] = (plateRBCorner.y - plateRTCorner.y) * tmp3f +
       plateRTCorner.y;   // RTRBy
20 tmp1v4f.val[3] = (float)(plateRBCorner.x - plateRTCorner.x) / (
       plateRBCorner.y - plateRTCorner.y) * (tmp1v4f.val[2] -
       plateRTCorner.y) + plateRTCorner.x;   // RTRBx
21 // pro vertikalni caru je logika obdobna, pouze bude otocena
       funkcni zavislost ... x  = f(y)
22 ⋮
```

Zdr. 18 Nalezení spojnic kuličky || s hranicemi nakloněné roviny

Sklon modrých spojnic protínající kuličku už nyní odpovídá reálné situaci, avšak stále neprotínají kuličku v jejím středu. Pozice spojnic jsou stále zatížené chybou vycházející z výpočtů převodových poměrů, které se počítají z vertikální spojnice LTRT a LBRB (Obr. 4.44, rovnice 4.3 a Zdr. 18, řádek 12 až 14) a horizontální spojnice LTLB a RTRB. Tyto ohraničení ale nejsou při náklonu roviny rovnoběžné, proto se převodový poměr mění a je nutné jej počítat přímo ze spojnice jdoucí přes kuličku. Velikost a sklon spojnice byl ale vypočten v před-

chozím kroku, takže lze zpřesnění provézt až nyní a nemůže se sloučit do jednoho výpočtu. Zjednodušeně řečeno, převodový poměr se musí počítat ve směru hran nakloněné roviny, protože jejich fyzická délka je známá a může se tedy dát do poměru - rovnice 4.4 a Zdr. 19. Zbývající část výpočtu je obdobná jako v rovnici 4.3 a Zdr. 18.

$$rat_y = \frac{\sqrt{(centerOfMass_x - LTRT_x)^2 + (centerOfMass_y - LTRT_y)^2}}{\sqrt{(LBRB_x - LTRT_x)^2 + (LBRB_y - LTRT_y)^2}} \tag{4.4}$$

```
// horizontalni cara - zpresneni vypoctu
tmp3f = sqrt(pow(centerOfMass.y - tmp2v4f.val[1], 2) + pow(
    centerOfMass.x - tmp2v4f.val[0], 2)) / sqrt(pow(tmp2v4f.val[3]
    - tmp2v4f.val[1], 2) + pow(tmp2v4f.val[2] - tmp2v4f.val[0], 2))
    ; // rat_y
```

Zdr. 19 Zpřesnění převodového poměru pro horizontální spojnici

Zjištění převodového poměru je nutné pro výpočet reálné pozice kuličky na nakloněné rovině - kapitola 4.7.5.

4.6.8 Záznam dráhy sledovaného objektu

Pro rychlou vizuální kontrolu kvality regulace byl přidán grafický záznam skutečné pozice kuličky v podobě křivky, která se v reálném čase vykresluje přímo do obrazu. Proto je nutné udržovat v paměti určitý počet (uživatelsky nastavitelný) pozic, ty průběžně doplňovat, přepisovat, mazat apod. - podle situace. Po dosažení nastaveného počtu pozic se bude trajektorie přepisovat, čím dále je pozice v minulosti, tím více se křivka ztenčuje, až zmizí úplně.
Žádaná délka sledovací čáry se mění v GUI. Pokud se tedy vyvolá změna, dojde k řízené změně velikosti vektoru, v kterém se uchovávají průběžné pozice kuličky. Zvětšení délky je jednoduché, vektor se pouze doplní do požadované velikosti. Při zmenšení se však musí smazat nejdříve hodnoty za aktuální pozicí ve vektoru a v případě dosažení konce vektoru se bude mazat od začátku, než bude dosaženo požadované délky - Zdr. 20.

```
// zmena velikosti vektoru z GUI - zvetseni
if(trackLineSizeSet > trackLineSize) {
    while(trackLine.size() < trackLineSizeSet) {
        trackLine.insert(trackLineCounter, -1);
        trackLineCounter++;
        trackLine.insert(trackLineCounter, -1);
        trackLineCounter++;
    }
trackLineSize = trackLineSizeSet;
}
// zmena velikosti vektoru z GUI - zmenseni
else if(trackLineSizeSet < trackLineSize) {
    while(trackLine.size() > trackLineSizeSet) {
        if(trackLine.size() > trackLineCounter) {
            trackLine.removeAt(trackLineCounter);
            trackLine.removeAt(trackLineCounter);
        }
        else {
            trackLine.removeFirst();
            trackLine.removeFirst();
            trackLineCounter -= 2;
        }
    }
trackLineSize = trackLineSizeSet;
}
```

Zdr. 20 Vykreslování dráhy sledovaného objektu - úprava velikosti

Pokud bude dosažena požadovaná velikost křivky, průběžně se přepisují „nejstarší“ prvky vektoru aktuálními pozicemi. Při dosažení konce vektoru se vynuluje počítadlo určující místo zápisu aktuální pozice a začíná se tedy přepisovat od začátku. Pokud není žádný objekt detekovaný, ukládá se do vektoru hodnota -1, která je při vykreslování detekovaná jako vadná a obchází se - Zdr. 21.

Příklad záznamu dráhy kuličky na reálném snímku je možné vidět na Obr. 4.35 a Obr. 4.36. Do každého snímku se postupně vykreslí všechny body zaznamenané ve vektoru a spojí se plnou čarou. Protože je vzdálenost mezi jednotlivými body minimální (dáno rychlostí zpracování obrazu), shluk úseček se jeví jako hladká křivka.

Do této chvíle se zpracování obrazu zaobíralo vesměs uživatelskou grafickou částí,

v které se využívaly zjištěné data z kalibrace a následných kinematických transformací. Ty se vykonají na začátku každého sejmutí snímku a jejich vysvětlením se zaobírá následující podkapitola.

```
if(trackLine.size() >= trackLineSize) {
    if(trackLineCounter >= trackLineSize) {
        trackLineCounter = 0;
    }
    // pokud je objekt detekovany, tak pridavam na nasledujici
        misto vektoru dalsi prvky
    if(objectDetected) {
        trackLine.replace(trackLineCounter, centerOfMass.x);
        trackLineCounter++;
        trackLine.replace(trackLineCounter, centerOfMass.y);
        trackLineCounter++;
    }
    // pokud detekovany neni, pridavam hodnotu -1 indikujici chybu
    else {
        trackLine.replace(trackLineCounter, -1);
        trackLineCounter++;
        trackLine.replace(trackLineCounter, -1);
        trackLineCounter++;
    }
}
```

Zdr. 21 Vykreslování dráhy sledovaného objektu - přepis prvků

4.7 Kinematické transformace

Analytická geometrie, která byla doposud vrcholem problematiky matematické části, se v této podkapitole rozšíří o kinematické transformace, nutné pro přepočet mezi 2D prostorem kamery a 3D prostorem modelu. Po vyřešení problematiky probírané v této podkapitole je možné zodpovědně říci, že bez maticových kinematických transformací je velice komplikované (ne-li nemožné), odvodit obecný matematický zápis rovnic vyjádřujících přepočty nutné pro následné řízení modelu. Metodám regulace je věnována samostatná podkapitola.

4.7.1 Kamerové vidění → reálné prostředí

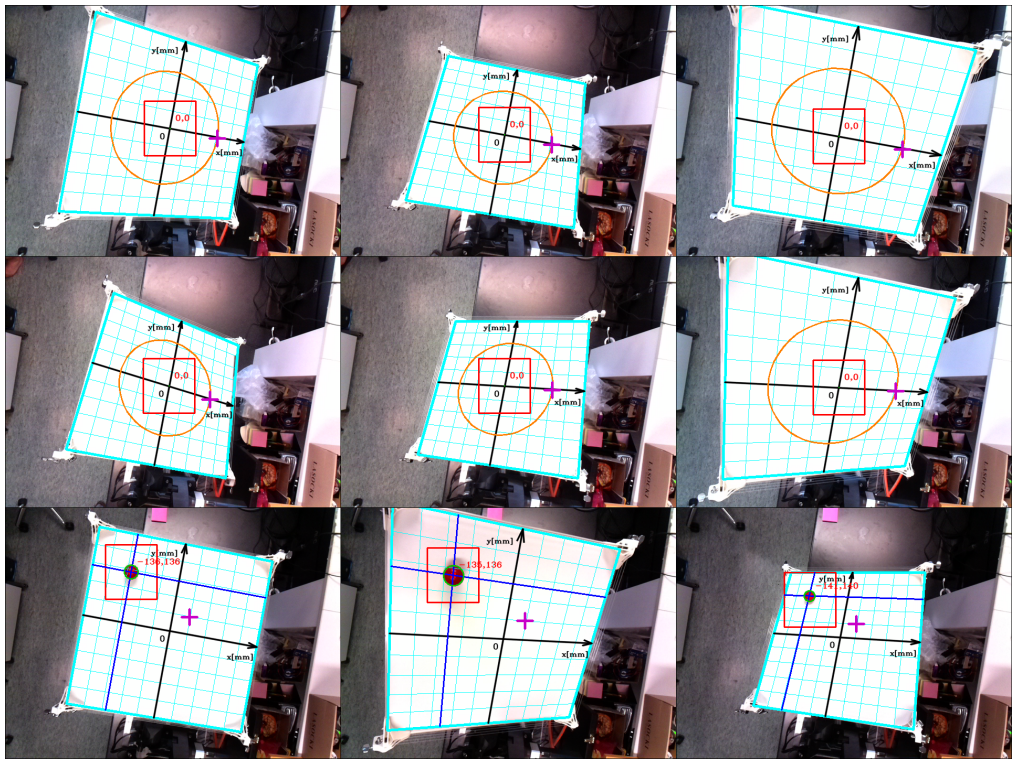
Sejmutý obraz z kamery nemusí odpovídat (a zpravidla nikdy přesně neodpovídá) reálné scéně. Tato nepříjemná skutečnost je dána kombinací rušivých elementů, vyplývajících jak z hardwarových vlastností použité kamery (geometrie a kvalita čočky, rozlišení čipu atd.), tak z fyzikální reality, kterou nelze obejít a musí se kompenzovat.

Pokud se konstrukční parametry kamery nemění (především čočka/objektiv zůstává stejný, nijak nedegraduje a je ve stejné pozici vůči čipu kamery), tak není příliš složité vady obrazu jednorázově kompenzovat. Při neznámé geometrii čočky se ve fázi kalibrace zjistí její zkreslení (s použitím obrazce známého tvaru přes celou snímanou plochu - většinou kalibrační šachovnice). Při porovnání známého obrazce se sejmutým snímkem se určí matice definující rozdíl snímku z kamery vůči reálné scéně a touto maticí (přesněji řečeno její inverzí) se každý sejmutý snímek vynásobí. Při předpokladu neměnné konfigurace mezi objektivem a čipem kamery jsou v této matici pouze konstanty, je po celou dobu zpracování neměnná a výsledkem je zpětné „narovnání“ obrazu. Ve většině případů (i v tomto) se používají širokoúhlé objektivy pro zaznamenání co možná největší plochy, při co možná nejmenší vzdálenosti kamery od snímané scény. Jedná se tedy o tzv. korekci soudkovitého zkreslení, postup je známý a cílem tohoto odstavce bylo pouze nastínit úlohu a princip korekční funkce. Použitá kamera má navíc tuto korekci implementovanou v knihovních funkcích pro sejmutí snímku, ve fázi předzpracování obrazu již tedy není nutné ji provádět.

Neporovnatelně větším problémem je ovšem druhá skutečnost - fyzikální re-

alita. V případě stacionárního umístění kamery a pohyblivé nakloněné rovině je potřeba vymyslet transformaci mezi 2D prostorem kamery a 3D prostorem reálné scény. Zde se v průběhu regulace mění náklon roviny, tedy je proměnné perspektivní zkreslení, kamera je umístěna mimo střed roviny, v neznámé vzdálenosti od ní, je vůči ní natočená, vzdálenost ohniska objektivu a roviny kamerového čipu zjištěná z datasheetu není správná (zjištěno experimentálním měřením - mění se s přítlakem optiky umístěné na molitanovém držáku), osy rotace roviny jsou zcela mimo optickou osu kamery apod. To znamená, že ze čtvercové roviny se stane z pohledu kamery konvexní různoběžník, který mění v prostoru jak svoji polohu a velikost, tak i orientaci. V podstatě jediné jednoznačně správné a známé parametry jsou fyzická velikost nakloněné roviny a fyzická velikost kamerového čipu. Z toho vyplývá, že je potřeba navrhnout, jak postupně určit a dopočítat všechny parametry nutné pro určení reálné pozice kuličky na nakloněné rovině, ze zjištěné pozice kuličky v obrazu.

Níže popsaný nalezený postup vyplývá z kinematického uspořádání modelu, umístění a orientace použité kamery a je nutné dodržet danou posloupnost operací. Neexistuje a nikdy nebude existovat univerzální strategie, která by vedla k těmto 2D $\rightarrow$ 3D přepočtům a mohla by být implementována v podobě knihovní funkce, jako je tomu např. u korekce zkreslení objektivu kamery. Je zde tolik proměnných a možností kinematického uspořádání, že zkrátka není možné vymyslet obecný postup. Uvedená strategie je odvozená speciálně pro tento model, nemusí být jediná a je průnikem mnoha pokusů, z nichž většina nevedla ke správnému řešení. Problematika totiž spadá do oblasti křivočarých prostorů s proměnnou křivostí všech třech dimenzí, což je autorovi zcela neznámá látka a musí tedy najít jiný způsob, jak tuto jistě složitou teorii obejít a jak postupně jednotlivé dimenze „narovnat“ a tím vyřešit zadaný technický problém. Všechny výpočty musí být samozřejmě zpracovávané v reálném čase na použitém hardware.
Na Obr. 4.45 jsou zobrazeny reálné snímky po aplikaci procedury, která bude popisována v dalších kapitolách. Jedná se o využití mezivýpočtů (vedoucích ke zjištění reálné pozice kuličky na nakloněné rovině), použitých pro vizuální prezentaci uživatelsky zajímavých dat (přehledné náčrty s postupným vykreslením jsou na Obr. 4.43 a Obr. 4.44). Vše se po transformacích jeví jako promítnuté na nakloněnou rovinu. Nejvíce patrná je asi změna tvaru a natočení elipsy zob-



Obr. 4.45 Reálné snímky z kamery znázorňující zkreslení obrazu

razující žádanou trajektorii. Pochopitelně se nejedná o elipsu, ale o kružnici, která se ovšem z pohledu kamery při náklonu roviny deformuje. Stejně tak černě zaznačené osy jsou na sebe kolmé a nakloněná rovina je tvaru čtverce. Vše je přes zpětné transformace přepočteno a poté vykresleno do obrazu, včetně sklonu šipek souřadné soustavy - nakloněná rovina je ve skutečnosti čistě bílá.

Poslední řada snímků na Obr. 4.45 zobrazuje fixní polohu kuličky při náklonu roviny do krajních poloh. V obraze je kulička vždy na zcela jiném místě a při ignorování dříve popisovaných skutečností by se skutečná poloha kuličky výrazně měnila. Fyzicky je ale na fixní pozici na rovině a i při náklonu by se její vypočítaná reálná poloha měnit neměla (a to je až na malou nepřesnost splněno - červeně vypsané souřadnice u kuličky). Nepřesnosti jsou způsobeny především použitím levné kamery, objektivem s nepřesnou geometrií u okrajů čočky, malým rozlišením čipu a zaokrouhlovací chybou. U náklonu roviny okolo

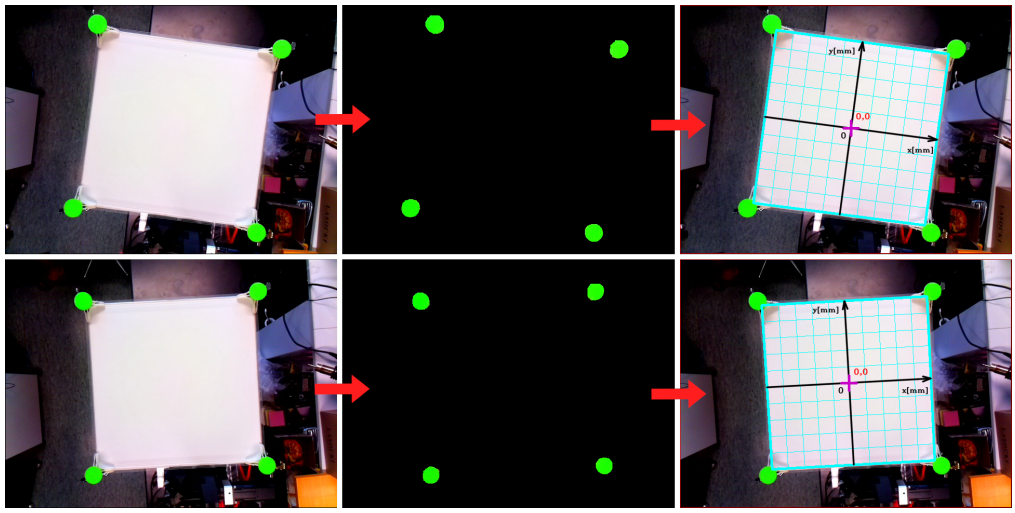
horizontální polohy, což je drtivá většina případů, se pohybuje chyba v řádech jednotek milimetrů, u náklonů blízkých fyzickým dorazům a pozici kuličky u okrajů roviny může chyba dosáhnout až k jednotkám centimetrů. To je však při porovnání případu bez průběžných transformací stále v únosných mezích, tam by byla chyba až desetinásobná - viz autorova publikace [P. 1] - článek navíc počítá se znatelně zjednodušenou situací, kde optická osa kamery prochází přes střed nakloněné roviny a kamera není vůči rovině nijak natočená. V článku je tedy demonstrovaný pouze vliv změny náklonu roviny na reálnou polohu kuličky. V případě této práce by byla chyba výpočtu výrazně větší.

4.7.2 Kalibrace rohů/ohraničení nakloněné roviny

Prvním a zásadním krokem je kalibrace rohů nakloněné roviny. Tento krok je klíčový jak pro zjištění základních parametrů jako je orientace kamery vůči rovině, vzdálenosti kamery od roviny, převodového poměru z pixelů na metry, tak pro uložení pozic rohů nakloněné roviny a jejího středu při nulovém náklonu (rovina v horizontální poloze). Tyto pozice jsou používané snad ve všech přepočtových vztazích, což vychází ze zvoleného postupu, který pro kinematické přepočty využívá matice homogenní transformace. Při změně orientace kamery je nutné kalibraci opakovat.

Samotná kalibrace má několik požadavků, které je pro správný výsledek nutné splnit. Prvním z nich je takové umístění kamery, aby byly v zorném poli všechny kalibrační obrazce (umístěné v rozích roviny). Dalším požadavkem je orientace kamery nepřekračující $\pm$ 45° vůči natočení nakloněné roviny (obraz rozdělen na čtyři kvadranty). Posledním požadavkem je zajištění, aby byla rovina při kalibraci v horizontální poloze.

S ohledem na použití algoritmů zpracování obrazu i pro kalibrační sekvenci musí být vypnuté sledování kuličky i detekce hran, oblast zájmu nastavená na celý snímek a barva sledovaného objektu na barvu kalibračních obrazců (v tomto případě zelená - dvojklik myší na kalibrační obrazec). Po nastavení vhodných světelných podmínek a kontrole, že bylo odfiltrováno vše kromě kalibračních obrazců, spustí uživatel kalibraci.
Na Obr. 4.46 je vidět postupná kalibrace na třech po sobě jdoucích snímcích.



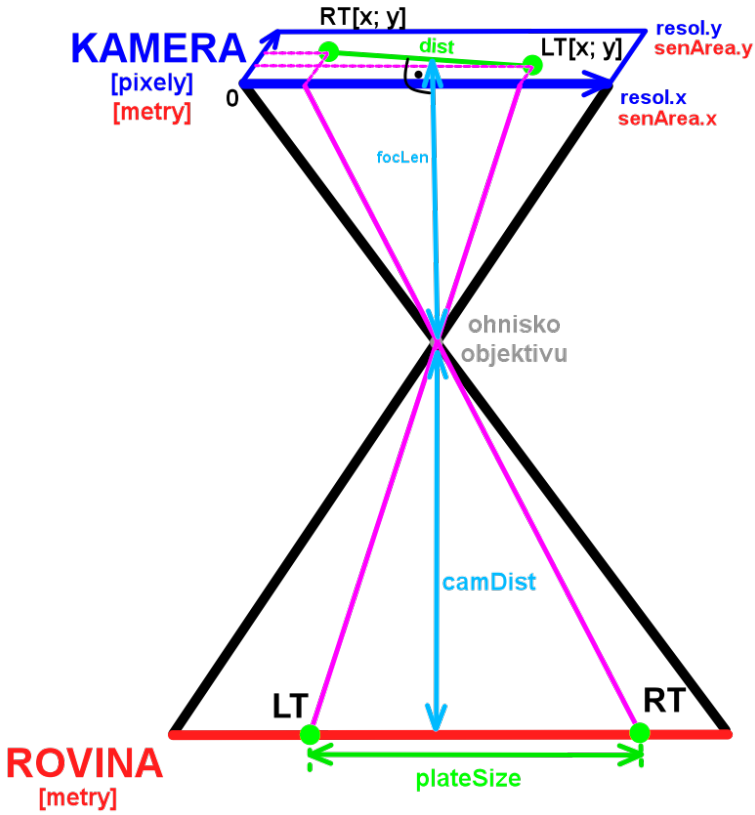
Obr. 4.46 Kalibrace nakloněné roviny

Nejdříve je obraz podroben algoritmům zpracování obrazu popisovaným v kapitole 4.6. Poté je vyfiltrovaný snímek rozdělen na čtyři v kvadranty, v každém je spočítáno plošné hmotnostní těžiště a nalezené souřadnice jsou uloženy. Aby souřadnice odpovídaly rohům nakloněné roviny, musí být provedena ještě korekce. Ta je nutná kvůli umístění kalibračních obrazců nad styčnou plochou roviny - čistým propojením nalezených bodů by měla hranice přesah, jak je možné si domyslet při pohledu na poslední snímky na Obr. 4.46. Příčinou je perspektivní zkreslení, jehož míra je závislá na vzdálenosti kamery od roviny.

Ze zjištěných pozic kalibračních obrazců jsou dále dopočítány další výchozí kinematické parametry. Ty se nemění, pokud se nezmění konstrukční parametry modelu, poloha a orientace kamery apod. Jejich nalezení je klíčové pro odvození kinematických transformačních matic a patří mezi ně pozice středu roviny v obrazu, úhlové natočení kamery vůči hranám roviny, převodový poměr mezi obrazovými pixely a reálnou velikostí v metrech a konečně skutečná vzdálenost ohniska kamery od nakloněné roviny (rovina v horizontální poloze).

Nejdříve je nutné vypočítat poslední uvedený parametr - vzdálenost kamery od nakloněné roviny, protože pouze pro odvozenou rovnici tohoto výpočtu jsou známé všechny proměnné. Nákres situace pro výpočet z jedné ze čtyř stran je uveden na Obr. 4.47 a odpovídají mu rovnice 4.3. V reálném programu probíhá

výpočet na všech čtyřech stranách roviny a výsledek se průměruje kvůli eliminaci nepřesností - Zdr. 22. Protože jsou kalibrační obrazce nad rovinou, je nutné ještě k výsledné vzdálenosti připočíst jejich odsazení, což je známý konstrukční parametr. Obraz přijímaný z kamery je již otočený správným směrem, tedy *LT* a *RT* je ve snímku orientováno správně, ne jako na Obr. 4.47. Při otočení by ale nebyl nákres názorný a z pohledu výpočtu se pouze otáčí znaménko, což je v pořádku, protože při umocňování na znaménku stejně nezáleží - výsledkem je vždy kladné číslo (určuje se velikost).



Obr. 4.47 Výpočet vzdálenosti ohniska kamery od nakloněné roviny

$$\frac{focLen}{dist} = \frac{camDist}{plateSize} \Rightarrow camDist = \frac{focLen \cdot plateSize}{dist}$$
$$camDist = \frac{focLen \cdot plateSize}{\sqrt{\left(\frac{RT_x - LT_x}{resol_x} \cdot\ senArea_x\right)^2 + \left(\frac{RT_y - LT_y}{resol_y} \cdot senArea_y\right)^2}} \tag{4.5}$$

```
camPlateDist =  calibCirclePlateDist + (
focalLength * plateSize.x / sqrt(pow((float)(plateRTCorner.x -
    plateLTCorner.x) / resolution.x * sensorArea.x, 2) + pow((float
    )(plateRTCorner.y - plateLTCorner.y) / resolution.y *
    sensorArea.y, 2))
+
⋮
)/4;
```

Zdr. 22 Výpočet vzdálenosti ohniska kamery od nakloněné roviny v programu

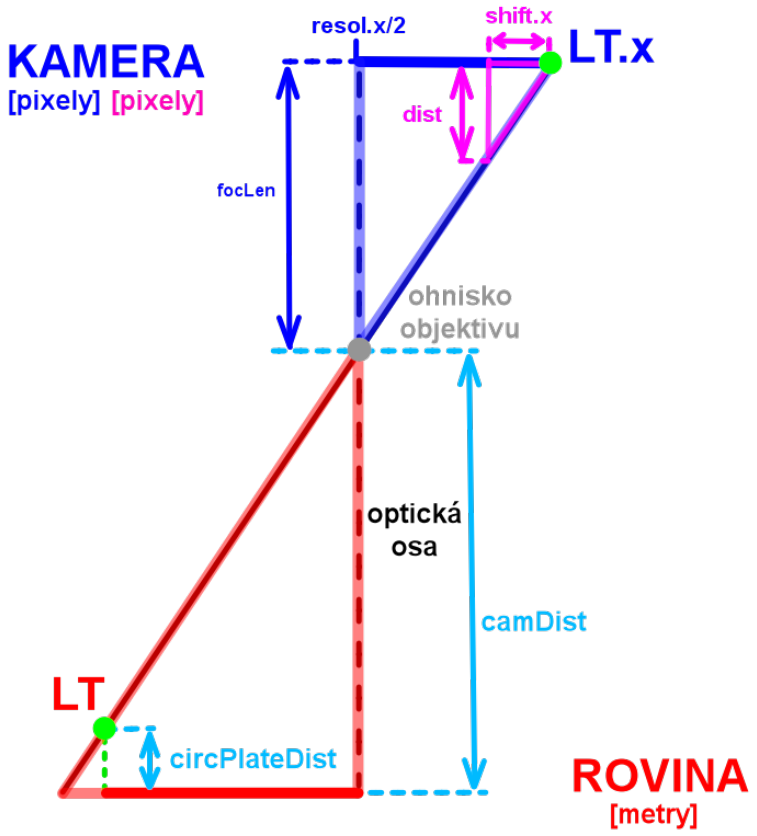
Ze zjištěné vzdálenosti kamery je možné dopočítat další uvedené parametry, z nichž prvním je korekce odsazení kalibračních obrazců v závislosti na perspektivním zkreslení kamery - Obr. 4.48. Stejně jako u předchozího nákresu je snímaný obraz z kamery již přetočený správným směrem, ale pro jednoduchost a srozumitelnost je vzorová situace a odvozený výpočet znázorněn bez tohoto přetočení.

$$\frac{shift_x}{\frac{resol_x}{2} - LT_x} = \frac{dist}{focLen} = \frac{circPlateDist}{camDist}$$
$$shift_x = \left(\frac{resol_x}{2} - LT_x\right) \cdot \frac{circPlateDist}{camDist} \tag{4.6}$$

```
plateLTCornerCalib = Point(plateLTCorner.x + (resolution.x/2 -
    plateLTCorner.x) * calibCirclePlateDist / camPlateDist,
    plateLTCorner.y + (resolution.y/2 - plateLTCorner.y) *
    calibCirclePlateDist / camPlateDist);
```

Zdr. 23 Přepočet pozice kalibračních obrazců v programu

Výpočty pro určení zbývajících parametrů už nejsou složité, proto bude využito nákresů na Obr. 4.47 a Obr. 4.48. Stále se však jedná o nepostradatelné parametry při dalších výpočtech, takže jejich určení je neméně důležité. Jsou jimi pozice středu nakloněné roviny z pohledu kamery, úhlové natočení roviny vůči optické



Obr. 4.48 Přepočet pozice kalibračních obrazců kvůli perspektivnímu zkreslení

ose kamery a převodový poměr z obrazových pixelů na metry. Pro zlepšení přesnosti výsledných výpočtů je použita metoda průměrování a pro jednoduchost je výpočet uvedený pouze pro osu x, pokud má daný parametr rovinné zobrazení.

$$
\begin{aligned}
plateCenter_x &= \frac{RT_x - LT_x + RB_x - LB_x}{4} + \frac{LB_x - LT_x}{2} + LT_x \\
plateRotShift &= atan\left(\frac{LT_y - RT_y}{RT_x - LT_x}\right) \\
px2mRat_x &= \frac{plateSize_x \cdot 2}{\sqrt{(RT_x - LT_x)^2 + (RT_y - LT_y)^2} + \sqrt{(RB_x - LB_x)^2 + (RB_y - LB_y)^2}}
\end{aligned}
\tag{4.7}
$$

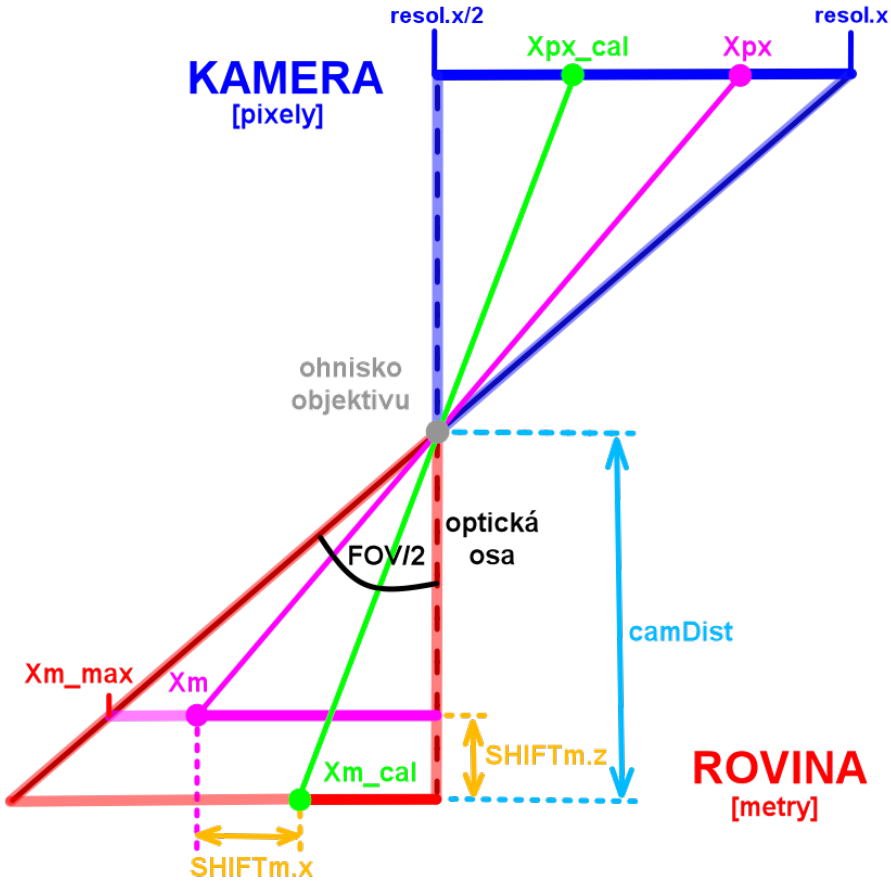
```
1 // ulozeni stredu naklonene roviny prumerovanim ohraniceni, pro
    pripad natoceni kamery
2 plateCenter.x = (plateRTCorner.x - plateLTCorner.x + plateRBCorner
    .x - plateLBCorner.x) / 4 + (plateLBCorner.x - plateLTCorner.x)
     / 2 + plateLTCorner.x;
3 // vypocet uhlove natoceni kamery vuci naklonene rovine
4 plateRotationShift = atan((float)(plateLTCornerCalib.y -
    plateRTCornerCalib.y) / (plateRTCornerCalib.x -
    plateLTCornerCalib.x));
5 // vypocet prevodoveho pomeru z pixelu na metry pri zname
    velikosti naklonene roviny
6 px2mRatio.x = (float)plateSize.x * 2 / (sqrt(pow(plateRTCorner.x -
     plateLTCorner.x, 2) + pow(plateRTCorner.y - plateLTCorner.y,
    2)) + sqrt(pow(plateRBCorner.x - plateLBCorner.x, 2) + pow(
    plateRBCorner.y - plateLBCorner.y, 2)));
```

Zdr. 24 Určení zbývajících parametrů modelu z výsledků kalibrace

Po provedení kalibrace a uložení výsledků do konfiguračního souboru (jeden stisk tlačítka; konfigurační soubor se načítá při startu aplikace) se již mohou kalibrační obrazce odmontovat. Přepočet ohraničení roviny během jejího náklonu se už dále dopočítává z kinematických transformací.

4.7.3 Využití transformačních matic pro online přepočet rohů/ohraničení roviny při jejím naklápění

Kinematické transformační matice budou sloužit k určení libovolného bodu v 3D prostoru, jehož původní pozice byla jednoznačně dána kinematickým uspořádáním modelu. Na základě změny úhlů náklonu roviny se bude s využitím transformačních matic dopočítávat posuv požadovaného bodu v 3D prostoru, oproti jeho pozici při nulovém náklonu roviny (horizontální poloha). Výsledná pozice bodu v kartézské soustavě souřadnic se musí až na výjimečné případy dále přepočítat do 2D pohledu kamerového čipu. Tento přepočet souvisí s již dříve zmíněným perspektivním zkreslením a ve všech případech je nutné jej na výsledné relativní posuny aplikovat. Názorná ukázka přepočtu pozice bodu pro jednu souřadnici, na základě vypočtených relativních posunů pomocí transformačních matic, je zobrazena na Obr. 4.49.



Obr. 4.49 Ukázka přepočtu x-ové souřadnice bodu z pohledu kamery při náklonu roviny

$$\frac{Xm_{max}}{Xm} = \frac{\frac{resol_x}{2}}{Xpx - \frac{resol_x}{2}} \Rightarrow Xpx = \frac{resol_x}{2} + \frac{\frac{resol_x}{2} \cdot Xm}{Xm_{max}}$$

$$Xm_{max} = tan\left(\frac{FOV}{2}\right) \cdot (camDist - SHIFTm_z)$$

$$Xm = Xm_{cal} + SHIFTm_x \tag{4.8}$$

$$Xm_{cal} = \left(Xpx_{cal} - \frac{resol_x}{2}\right) \cdot px2mRat_x$$

$$Xpx = \frac{resol_x}{2} + \frac{resol_x \cdot \left[\left(Xpx_{cal} - \frac{resol_x}{2}\right) \cdot px2mRat_x + SHIFTm_x\right]}{2 \cdot tan\left(\frac{FOV}{2}\right) \cdot (camDist - SHIFTm_z)}$$

Jako příklad je zde uvedený výpočet pouze pro aktualizaci *x-ové* pozice pravého horního rohu roviny při jejím náklonu. Pro jakýkoliv jiný bod je výpočet obdobný, změní se pouze názvy odpovídajících parametrů zjištěných při kalibraci (výchozí pozice kalibračních obrazců). Přesnost výpočtu je subpixelární, pozice v obrazu se proto zaokrouhluje na celý pixel.

```
plateRTCorner.x = round(resolution.x/2 + resolution.x * ((
    plateRTCornerCalib.x - resolution.x/2) * px2mRatio.x +
    plateRTCornerShift.x) / (2 * tan(FOV.x/2) * (camPlateDist -
    plateRTCornerShift.z)));
```

Zdr. 25 Aktualizace pozice pravého horního rohu roviny při změně náklonu

Ve výsledné rovnici 4.8 jsou stále však 2 neznámé parametry (posun oproti výchozí pozici - $SHIFTm$), které je nutné nějakým způsobem dopočítat. K jejich určení byl zvolený přístup s využitím kinematických transformačních matic, jejichž odvozením se zabývá následující podkapitola. Postup, jakým byl kýžený posuv v ose X/Y a Z určený, demonstruje rovnice 4.9.

$$\overrightarrow{SHIFTm} = {}^{G}\vec{r} - {}^{G}\vec{r}_{Calib}$$

$${}^{G}\vec{r} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix}_{\begin{pmatrix} x = x_0 \\ y = y_0 \\ \varphi_X = 0 \\ \varphi_Y = 0 \\ bR = z_0 \end{pmatrix}} ; \quad {}^{G}\vec{r}_{Calib} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix}_{\begin{pmatrix} \alpha = 0 \\ \beta = 0 \\ x = x_0 \\ y = y_0 \\ \varphi_X = 0 \\ \varphi_Y = 0 \\ bR = z_0 \end{pmatrix}} \tag{4.9}$$

Konkrétní výpočet posuvu, např. opět pro pravý horní roh nakloněné roviny RT, je pro představu uvedený v rovnici 4.10.

$$\overrightarrow{SHIFTm}_{(RT)} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix}_{\begin{pmatrix} x = plateSize_x/2 \\ y = plateSize_y/2 \\ \varphi_X = 0 \\ \varphi_Y = 0 \\ bR = 0 \end{pmatrix}} - \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix}_{\begin{pmatrix} \alpha = 0 \\ \beta = 0 \\ x = plateSize_x/2 \\ y = plateSize_y/2 \\ \varphi_X = 0 \\ \varphi_Y = 0 \\ bR = 0 \end{pmatrix}} \tag{4.10}$$

Výsledný tvar rovnice 4.10 pro konkrétní kinematické uspořádání modelu je uveden ve Zdr. 26. Jedná se o poslední neznámé parametry ze soustavy rovnic, jejichž hledání započalo v kapitole 4.6.5.

```
1 plateRTCornerShift.x = 0.5845*sin(plateRotationShift) - 0.25*cos(
    plateRotationShift) + 0.25*cos(beta)*cos(plateRotationShift) -
    0.5845*cos(alfa)*sin(plateRotationShift) + 0.25*sin(alfa)*sin(
    beta)*sin(plateRotationShift);
2 plateRTCornerShift.y = 0.5845*cos(plateRotationShift) + 0.25*sin(
    plateRotationShift) - 0.25*cos(beta)*sin(plateRotationShift) -
    0.5845*cos(alfa)*cos(plateRotationShift) + 0.25*cos(
    plateRotationShift)*sin(alfa)*sin(beta);
3 plateRTCornerShift.z = 0.5845*sin(alfa) + 0.25*cos(alfa)*sin(beta)
    ;
```

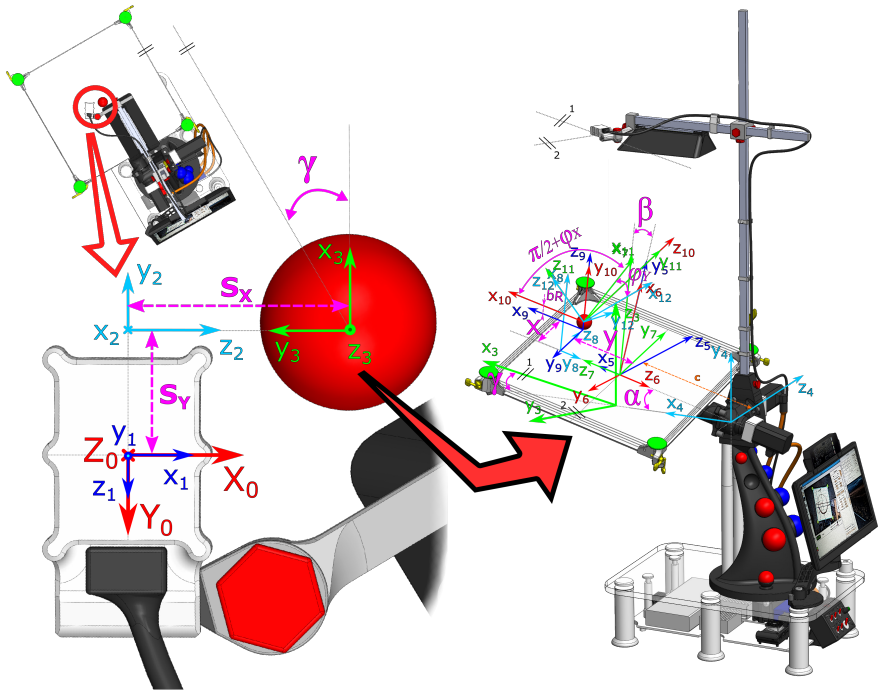
Zdr. 26 Výpočet posuvu rohu roviny RT v absolutních souřadnicích

4.7.4 Odvození kinematických transformačních matic

Pomocí matice homogenní transformace je možné, mimo jiného, určit pozici bodu v globální souřadné soustavě (dále GSS) při známé změně stavových parametrů modelu. Určení transformační matice musí odpovídat konkrétnímu zadání, proto bylo toto zadání upřesněno nejdříve v předchozích kapitolách, a až nyní bude probíhat její odvození. Aby byla transformační matice pokud možno co nejvíce univerzální, tedy aby nebylo nutné pro každý bod provádět nové odvození, byla umístěna výchozí GSS do středu nakloněné roviny v její výchozí pozici a zarovnána s kamerou. Kamera totiž zůstává po jejím zafixování stacionární (důvod zarovnání GSS s kamerovým čipem), a měření pozice kuličky, vycházející z implementace navržených algoritmů zpracování obrazu, je vztažené vůči středu nakloněné roviny (Obr. 4.45), jehož pozice v obrazu už byla navíc i určena (Rce. 4.7). Proto je toto umístění GSS logické, na orientaci os nijak zvlášť nezáleží, protože souřadné soustavy (dále SS) kamery i roviny jsou orientovány rozdílně a přepočet tedy bude tak jako tak nutný. Protože se však používá častěji přepočet na SS kamery, byly osy orientovány stejně, jak je indexovaný sejmutý obraz z kamery. Posuv mezi SS kamery a nakloněné roviny (optická osa kamery neprotíná střed nakloněné roviny) je již promítnutý do zjištěné pozice rohů ro-

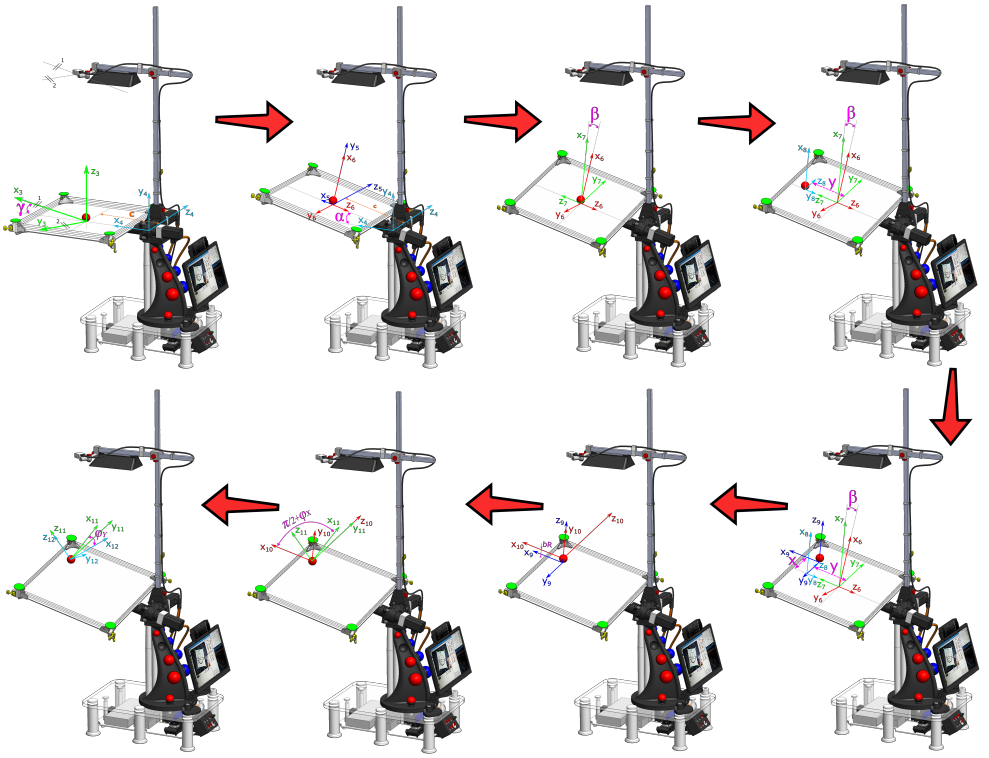
viny při kalibraci, takže se ve výsledných rovnicím neuplatní, jak bude vidět dále. Kvůli dříve zmíněné univerzálnosti transformační matice byly však ve výpočtu uvažovány všechny eventuality.

Pro určení matice homogenní transformace byla zvolena Denavit-Hartenbergova metoda rozmístění souřadných soustav (dále D-H notace) - [79], pomocí které se postupnými transformacemi odvodí matice popisující vztah mezi polohovým vektorem poslední LSS a polohovým vektorem GSS, na základě změny stavových parametrů modelu. Jinými slovy je možné zjistit, jak se bude měnit pozice i orientace libovolného bodu spojeného s modelem nakloněné roviny, při změně úhlů náklonu, pozice a orientace kamery apod., vůči pevné SS (v tomto případě spojené s kamerou).



Obr. 4.50 Rozmístění SS splňující požadavky vyplývající z D-H notace

Na Obr. 4.50 je kompletní rozmístění SS tak, aby byly důsledně splněny všechny požadavky D-H notace. Pro názornější a přehlednější kroky jednotlivých transformací (řádky v tabulce D-H parametrů) je zde Obr. 4.51.



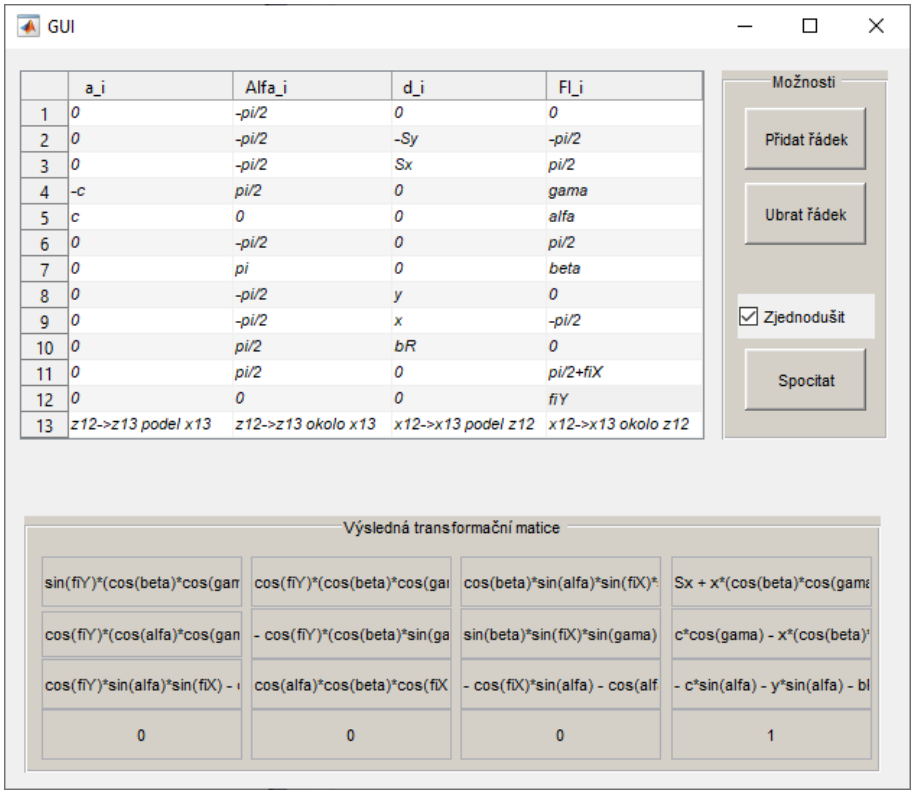
Obr. 4.51 Postupné rozmístění SS podle změny stavových parametrů modelu

Na základě rozmístění SS v Obr. 4.50 a Obr. 4.51 byla vytvořena tabulka D-H parametrů, pomocí které se následně určila výsledná matice homogenní transformace z LSS do GSS.

Pro přehledné vyplňování tabulky D-H parametrů, automatizování maticových přepočtů a využití funkcí pro zjednodušení transformační matice byla vytvořena GUI aplikace přímo v MATLABu, kde se provádějí veškeré výpočty a přípravy rovnic do tvaru vhodného pro řídicí aplikaci. Náhled pomocné GUI aplikace a zároveň vyplněná tabulka D-H parametrů je na Obr. 4.52.

V rovnici 4.11 jsou uvedené obecné i konkrétní vztahy pro výpočet matice homogenní transformace. Její přesný tvar je dosti komplexní a je uveden v rovnici 4.12. Uvedená transformační matice se používá k výpočtu stěžejních částí výsledků zpracování obrazu (např. v rovnici 4.8 a 4.15) a v podstatě u všech vykreslovaných prvků v obrazu řídicí aplikace (výsledky lze vidět např. na Obr. 4.45).

Je tedy klíčovou součástí celé sekvence operací, vedoucí k přepočtu mezi kamerovým viděním a polohou v reálném prostředí.



Obr. 4.52 Pomocná GUI aplikace v MATLABu - tabulka D-H parametrů

$$
{}^{i-1}T_i = \begin{bmatrix} \cos(\theta_i) & -\cos(\alpha_i)\sin(\theta_i) & \sin(\alpha_i)\sin(\theta_i) & a_i\cos(\theta_i) \\ \sin(\theta_i) & \cos(\alpha_i)\cos(\theta_i) & -\sin(\alpha_i)\cos(\theta_i) & a_i\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}
$$

$$
{}^{0}T_n = {}^{0}T_1 \cdot \ldots \cdot {}^{n-2}T_{n-1} \cdot {}^{n-1}T_n
$$

$$
{}^{0}T_{10} = {}^{0}T_1 \cdot {}^{1}T_2 \cdot {}^{2}T_3 \cdot {}^{3}T_4 \cdot {}^{4}T_5 \cdot {}^{5}T_6 \cdot {}^{6}T_7 \cdot {}^{7}T_8 \cdot {}^{8}T_9 \cdot {}^{9}T_{10} \cdot {}^{10}T_{11} \cdot {}^{11}T_{12}
$$

$$
{}^{0}T_{12} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & X_0 \\ r_{21} & r_{22} & r_{23} & Y_0 \\ r_{31} & r_{32} & r_{33} & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
\begin{aligned}
r_{11} = {} & \sin(\varphi_Y)\,[\cos(\beta)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma)] + \cos(\varphi_Y)\,[\cos(\alpha)\sin(\varphi_X)\sin(\gamma) - \\
& - \cos(\varphi_X)\cos(\gamma)\sin(\beta) + \cos(\beta)\cos(\varphi_X)\sin(\alpha)\sin(\gamma)] \\
r_{12} = {} & \cos(\varphi_Y)\,[\cos(\beta)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma)] - \sin(\varphi_Y)\,[\cos(\alpha)\sin(\varphi_X)\sin(\gamma) - \\
& - \cos(\varphi_X)\cos(\gamma)\sin(\beta) + \cos(\beta)\cos(\varphi_X)\sin(\alpha)\sin(\gamma)] \\
r_{13} = {} & \cos(\beta)\sin(\alpha)\sin(\varphi_X)\sin(\gamma) - \cos(\gamma)\sin(\beta)\sin(\varphi_X) - \cos(\alpha)\cos(\varphi_X)\sin(\gamma) \\
r_{21} = {} & \cos(\varphi_Y)\,[\cos(\alpha)\cos(\gamma)\sin(\varphi_X) + \cos(\varphi_X)\sin(\beta)\sin(\gamma) + \\
& + \cos(\beta)\cos(\varphi_X)\cos(\gamma)\sin(\alpha)] - \sin(\varphi_Y)\,[\cos(\beta)\sin(\gamma) - \cos(\gamma)\sin(\alpha)\sin(\beta)] \\
r_{22} = {} & - \cos(\varphi_Y)\,[\cos(\beta)\sin(\gamma) - \cos(\gamma)\sin(\alpha)\sin(\beta)] - \sin(\varphi_Y)\,[\cos(\alpha)\cos(\gamma)\sin(\varphi_X) + \\
& + \cos(\varphi_X)\sin(\beta)\sin(\gamma) + \cos(\beta)\cos(\varphi_X)\cos(\gamma)\sin(\alpha)] \\
r_{23} = {} & \sin(\beta)\sin(\varphi_X)\sin(\gamma) - \cos(\alpha)\cos(\varphi_X)\cos(\gamma) + \cos(\beta)\cos(\gamma)\sin(\alpha)\sin(\varphi_X) \\
r_{31} = {} & \cos(\varphi_Y)\sin(\alpha)\sin(\varphi_X) - \cos(\alpha)\sin(\beta)\sin(\varphi_Y) - \cos(\alpha)\cos(\beta)\cos(\varphi_X)\cos(\varphi_Y) \\
r_{32} = {} & \cos(\alpha)\cos(\beta)\cos(\varphi_X)\sin(\varphi_Y) - \sin(\alpha)\sin(\varphi_X)\sin(\varphi_Y) - \cos(\alpha)\cos(\varphi_Y)\sin(\beta) \\
r_{33} = {} & - \cos(\varphi_X)\sin(\alpha) - \cos(\alpha)\cos(\beta)\sin(\varphi_X) \\
X_0 = {} & S_x + x\,[\cos(\beta)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma)] + c\sin(\gamma) - bR\,[\cos(\gamma)\sin(\beta) - \\
& - \cos(\beta)\sin(\alpha)\sin(\gamma)] - c\cos(\alpha)\sin(\gamma) - y\cos(\alpha)\sin(\gamma) \\
Y_0 = {} & c\cos(\gamma) - x\,[\cos(\beta)\sin(\gamma) - \cos(\gamma)\sin(\alpha)\sin(\beta)] - S_y + bR\,[\sin(\beta)\sin(\gamma) + \\
& + \cos(\beta)\cos(\gamma)\sin(\alpha)] - c\cos(\alpha)\cos(\gamma) - y\cos(\alpha)\cos(\gamma) \\
Z_0 = {} & -c\sin(\alpha) - y\sin(\alpha) - bR\cos(\alpha)\cos(\beta) - x\cos(\alpha)\sin(\beta)
\end{aligned}
\tag{4.12}
$$

Rovnice 4.13 a 4.14 ukazuje použití transformační matice pro zjištění pozice poslední LSS v GSS, tedy pozice žádaného bodu v GSS, při libovolné konfiguraci nakloněné roviny i kamery.

Změnou parametrů x a y a vynulováním bR se určí libovolný bod na nakloněné rovině, což je první případ použití - zjištění pozice rohů roviny při jejím náklonu v GSS - ukázka na konci předchozí kapitoly.

$$
{}^{G}\vec{r} = {}^{G}T_B \cdot {}^{B}\vec{r} \quad \Rightarrow \quad \begin{bmatrix} X_P \\ Y_P \\ Z_P \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{G}R_B & & {}^{G}\vec{d} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \cdot \begin{bmatrix} x_P \\ y_P \\ z_P \\ 1 \end{bmatrix} \tag{4.13}
$$

$$ {}^0\vec{r} = {}^0T_{12} \,.^0\vec{r} \quad \Rightarrow \quad \begin{bmatrix} X_P \\ Y_P \\ Z_P \\ 1 \end{bmatrix} = {}^0T_{12} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \tag{4.14} $$

4.7.5 Využití transformačních matic pro určení reálné pozice kuličky na nakloněné rovině

Při určení pozice rohů při náklonu roviny, byla použita transformační matice pro zjištění posuvu (*SHIFTm*) v absolutních souřadnicích, oproti výchozí poloze rohů při kalibraci. Byla tedy k dispozici známá poloha v LSS (pozice rohů od středu nakloněné roviny - viz rovnice 4.10) a zjišťovala se poloha v GSS. U pozice kuličky vyvstává opačný problém: je známá její poloha v GSS (v ose X a Y, osa Z je neznámá, protože kamera poskytuje pouze 2D obraz) a je nutné zjistit její pozici v LSS (pozice na nakloněné rovině).

$$ Bx_{px} = \frac{resol_x}{2} + \frac{resol_x \cdot \left[\left(PlateCenterCal_x - \frac{resol_x}{2} \right) \cdot px2mRat_x + SHIFTm_{Bx} \right]}{2 \cdot tan\left(\frac{FOV}{2}\right) \cdot (camDist - SHIFTm_{Bz})} $$

$$ By_{px} = \frac{resol_y}{2} + \frac{resol_y \cdot \left[\left(PlateCenterCal_y - \frac{resol_y}{2} \right) \cdot px2mRat_y + SHIFTm_{By} \right]}{2 \cdot tan\left(\frac{FOV}{2}\right) \cdot (camDist - SHIFTm_{Bz})} $$

$$ \begin{aligned} SHIFTm_{Bx} = \; & (0.3345 \cdot \sin\gamma - bR \cdot (\cos\gamma \cdot \sin\beta - \cos\beta \cdot \sin\alpha \sin\gamma) + \\ & Bx_m \cdot (\cos\beta \cos\gamma + \cdot \sin\alpha \cdot \sin\beta \cdot \sin\gamma) - \\ & 0.3345 \cdot \cos\alpha \cdot \sin\gamma - By_m \cdot \cos\alpha \cdot \sin\gamma) \\ SHIFTm_{By} = \; & (0.3345 \cdot \cos\gamma + bR \cdot (\sin\gamma \cdot \sin\beta + \cos\beta \cdot \sin\alpha \cos\gamma) - \\ & Bx_m \cdot (\cos\beta \sin\gamma - \cdot \sin\alpha \cdot \sin\beta \cdot \cos\gamma) - \\ & 0.3345 \cdot \cos\alpha \cdot \cos\gamma - By_m \cdot \cos\alpha \cdot \cos\gamma) \\ SHIFTm_{Bz} = \; & (0.3345 \cdot \sin\alpha - bR + By_m \cdot \sin\alpha + Bx_m \cdot \cos\alpha \cdot \sin\beta + \\ & bR \cdot \cos\alpha \cdot \cos\beta) \end{aligned} \tag{4.15} $$

K tomu by bylo možné s výhodou použít inverzní transformaci, i když je jeden z parametrů neznámý (pozice v ose Z v GSS) - v LSS je pozice v ose z známá

(bR) a s její pomocí je dosaženo ve výsledku 2 rovnic o 2 neznámých. Problém je ovšem v tom, že je zjištěná pozice kuličky z kamery v GSS zatížena perspektivním zkreslením a pokud není k dispozici její vzdálenost od kamery (osa Z v GSS), není možné určit míru perspektivního zkreslení a tím v konečném důsledku ani její skutečnou pozici na nakloněné rovině. Je tedy nutné opět využít některý z již odvozených vztahů založených na známé pozici bodu v GSS při kalibraci, od kterého se pomocí transformační matice určí posun. To však nic nemění na skutečnosti, že úkol je oproti dosavadnímu řešení stále obrácený - je potřeba určit pozici kuličky v LSS, ne v GSS, jak jsou rovnice postaveny. Je tedy nutné vyjádřit z obou výsledných rovnic v 4.15 (pro X a Y pozici kuličky v GSS), neznámou pozici v ose x a y v LSS (Bx_m a By_m). Pro nalezení rovnice 4.15 byl použitý obdobný postup jako v rovnicích 4.10 a 4.8.

```
Bx_m  = (centerOfMass.x*px2mRatio.y*pow(resolution.y,2)*tanFOV2.x*sinAlfa - (669*centerOfMass.x*resolution.y*tanFOV2.x*cosGama*sinAlfa)/1000 -
         centerOfMass.y*px2mRatio.x*pow(resolution.x,2)*tanFOV2.y*sinAlfa + (669*centerOfMass.y*resolution.x*tanFOV2.y*sinAlfa*sinGama)/1000 +
         bR*resolution.x*resolution.y*cosAlfa*sinBeta + (669*resolution.x*resolution.y*tanFOV2.x*cosGama*sinAlfa)/2000 +
         (px2mRatio.x*pow(resolution.x,2)*resolution.y*tanFOV2.y*sinAlfa)/2 - (px2mRatio.y*resolution.x*pow(resolution.y,2)*tanFOV2.x*sinAlfa)/2 -
         (669*resolution.x*resolution.y*tanFOV2.y*sinAlfa*sinGama)/2000 + (px2mRatio.x*pow(resolution.x,2)*resolution.y*cosAlfa*cosGama)/2 -
         (px2mRatio.y*resolution.x*pow(resolution.y,2)*cosAlfa*sinGama)/2 - 2*centerOfMass.x*plateCenterCalib.y*px2mRatio.y*resolution.y*tanFOV2.x*sinAlfa +
         2*centerOfMass.y*plateCenterCalib.x*px2mRatio.x*resolution.x*tanFOV2.y*sinAlfa - plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*tanFOV2.y*sinAlfa +
         plateCenterCalib.y*px2mRatio.y*resolution.x*resolution.y*tanFOV2.x*sinAlfa + 2*centerOfMass.x*bR*resolution.y*tanFOV2.x*cosAlfa*cosGama -
         2*centerOfMass.x*bR*resolution.y*tanFOV2.x*cosBeta*cosGama + 2*centerOfMass.x*camPlateDist*resolution.y*tanFOV2.x*cosAlfa*cosGama -
         2*centerOfMass.y*bR*resolution.x*tanFOV2.y*cosAlfa*sinGama + 2*centerOfMass.y*bR*resolution.x*tanFOV2.y*cosBeta*sinGama -
         2*centerOfMass.y*camPlateDist*resolution.x*tanFOV2.y*cosAlfa*sinGama - bR*resolution.x*resolution.y*tanFOV2.x*cosAlfa*cosGama +
         bR*resolution.x*resolution.y*tanFOV2.x*cosBeta*cosGama - camPlateDist*resolution.x*resolution.y*tanFOV2.x*cosAlfa*cosGama -
         plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*cosAlfa*cosGama + bR*resolution.x*resolution.y*tanFOV2.y*cosAlfa*sinGama -
         bR*resolution.x*resolution.y*tanFOV2.x*cosBeta*sinGama + camPlateDist*resolution.x*resolution.y*tanFOV2.y*cosAlfa*sinGama +
         plateCenterCalib.y*px2mRatio.y*resolution.x*resolution.y*cosAlfa*sinGama - 2*centerOfMass.y*bR*resolution.x*tanFOV2.y*cosGama*sinAlfa*sinBeta -
         2*centerOfMass.x*bR*resolution.y*tanFOV2.x*sinAlfa*sinBeta*sinGama + bR*resolution.x*resolution.y*tanFOV2.y*cosGama*sinAlfa*sinBeta +
         bR*resolution.x*resolution.y*tanFOV2.x*sinAlfa*sinBeta*sinGama)/
        (resolution.x*resolution.y*cosAlfa*cosBeta + 2*centerOfMass.x*resolution.y*tanFOV2.x*cosGama*sinBeta - 2*centerOfMass.y*resolution.x*tanFOV2.y*sinBeta*sinGama -
         resolution.x*resolution.y*tanFOV2.x*cosGama*sinBeta + resolution.x*resolution.y*tanFOV2.y*sinBeta*sinGama +
         resolution.x*resolution.y*tanFOV2.y*cosBeta*cosGama*sinAlfa + resolution.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama -
         2*centerOfMass.y*resolution.x*tanFOV2.y*cosBeta*cosGama*sinAlfa - 2*centerOfMass.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama);

By_m  = ((669*resolution.x*resolution.y*cosBeta)/2000 + bR*resolution.x*resolution.y*sinAlfa - (669*resolution.x*resolution.y*cosAlfa*cosBeta)/2000 -
         (669*centerOfMass.x*resolution.y*tanFOV2.x*cosGama*sinBeta)/1000 + (669*centerOfMass.y*resolution.x*tanFOV2.y*sinBeta*sinGama)/1000 +
         (669*resolution.x*resolution.y*tanFOV2.x*cosGama*sinBeta)/2000 - (669*resolution.x*resolution.y*tanFOV2.y*sinBeta*sinGama)/2000 -
         (px2mRatio.y*resolution.x*pow(resolution.y,2)*cosBeta*cosGama)/2 - (px2mRatio.x*pow(resolution.x,2)*resolution.y*cosBeta*sinGama)/2 -
         (669*resolution.x*resolution.y*tanFOV2.x*cosAlfa*cosGama*sinBeta)/2000 - (669*resolution.x*resolution.y*tanFOV2.y*cosBeta*cosGama*sinAlfa)/2000 -
         (px2mRatio.x*pow(resolution.x,2)*resolution.y*tanFOV2.y*cosAlfa*sinBeta)/2 + (px2mRatio.y*resolution.x*pow(resolution.y,2)*tanFOV2.x*cosAlfa*sinBeta)/2 -
         (669*resolution.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama)/2000 + (669*resolution.x*resolution.y*tanFOV2.y*cosAlfa*sinBeta*sinGama)/2000 +
         2*centerOfMass.y*bR*resolution.x*tanFOV2.y*cosAlfa*cosGama - 2*centerOfMass.y*bR*resolution.x*tanFOV2.y*cosBeta*cosGama -
         2*centerOfMass.y*camPlateDist*resolution.x*tanFOV2.y*cosBeta*cosGama + (px2mRatio.x*pow(resolution.x,2)*resolution.y*cosGama*sinAlfa*sinBeta)/2 +
         2*centerOfMass.x*bR*resolution.y*tanFOV2.x*cosAlfa*sinGama - 2*centerOfMass.x*bR*resolution.y*tanFOV2.x*cosBeta*sinGama -
         2*centerOfMass.x*camPlateDist*resolution.y*tanFOV2.x*cosBeta*sinGama - (px2mRatio.y*resolution.x*pow(resolution.y,2)*sinAlfa*sinBeta*sinGama)/2 -
         bR*resolution.x*resolution.y*tanFOV2.y*cosAlfa*cosGama + bR*resolution.x*resolution.y*tanFOV2.y*cosBeta*cosGama +
         camPlateDist*resolution.x*resolution.y*tanFOV2.y*cosBeta*cosGama + plateCenterCalib.y*px2mRatio.y*resolution.x*resolution.y*cosBeta*cosGama -
         bR*resolution.x*resolution.y*tanFOV2.x*cosAlfa*sinGama + bR*resolution.x*resolution.y*tanFOV2.y*cosBeta*sinGama +
         camPlateDist*resolution.x*resolution.y*tanFOV2.x*cosBeta*sinGama + plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*cosBeta*sinGama +
         (669*centerOfMass.x*resolution.y*tanFOV2.x*cosAlfa*cosGama*sinBeta)/1000 + (669*centerOfMass.y*resolution.x*tanFOV2.y*cosBeta*cosGama*sinAlfa)/1000 -
         centerOfMass.x*px2mRatio.y*pow(resolution.y,2)*tanFOV2.x*cosAlfa*sinBeta + centerOfMass.y*px2mRatio.x*pow(resolution.x,2)*tanFOV2.y*cosAlfa*sinBeta +
         (669*centerOfMass.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama)/1000 - (669*centerOfMass.y*resolution.x*tanFOV2.y*cosAlfa*sinBeta*sinGama)/1000 +
         2*centerOfMass.x*plateCenterCalib.y*px2mRatio.y*resolution.y*tanFOV2.x*cosAlfa*sinBeta -
         2*centerOfMass.y*plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*tanFOV2.y*cosAlfa*sinBeta +
         plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*tanFOV2.y*cosAlfa*sinBeta -
         plateCenterCalib.y*px2mRatio.y*resolution.x*resolution.y*tanFOV2.x*cosAlfa*sinBeta +
         2*centerOfMass.x*bR*resolution.y*tanFOV2.x*cosGama*sinAlfa*sinBeta + 2*centerOfMass.x*camPlateDist*resolution.y*tanFOV2.x*cosGama*sinAlfa*sinBeta -
         2*centerOfMass.y*bR*resolution.x*tanFOV2.y*sinAlfa*sinBeta*sinGama - 2*centerOfMass.y*camPlateDist*resolution.x*resolution.y*tanFOV2.y*sinAlfa*sinBeta*sinGama +
         bR*resolution.x*resolution.y*tanFOV2.x*cosGama*sinAlfa*sinBeta - camPlateDist*resolution.x*resolution.y*tanFOV2.x*cosGama*sinAlfa*sinBeta -
         plateCenterCalib.x*px2mRatio.x*resolution.x*resolution.y*cosGama*sinAlfa*sinBeta + bR*resolution.x*resolution.y*tanFOV2.y*sinAlfa*sinBeta*sinGama +
         camPlateDist*resolution.x*resolution.y*tanFOV2.y*sinAlfa*sinBeta*sinGama + plateCenterCalib.y*px2mRatio.y*resolution.x*resolution.y*sinAlfa*sinBeta*sinGama)/
        (resolution.x*resolution.y*cosAlfa*cosBeta + 2*centerOfMass.x*resolution.y*tanFOV2.x*cosGama*sinBeta - 2*centerOfMass.y*resolution.x*tanFOV2.y*sinBeta*sinGama -
         resolution.x*resolution.y*tanFOV2.x*cosGama*sinBeta + resolution.x*resolution.y*tanFOV2.y*sinBeta*sinGama +
         resolution.x*resolution.y*tanFOV2.y*cosBeta*cosGama*sinAlfa + resolution.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama -
         2*centerOfMass.y*resolution.x*tanFOV2.y*cosBeta*cosGama*sinAlfa - 2*centerOfMass.x*resolution.y*tanFOV2.x*cosBeta*sinAlfa*sinGama);
```

Obr. 4.53 Výsledné rovnice určující reálnou pozici kuličky na nakloněné rovině

Výsledné rovnice určující reálnou pozici kuličky na nakloněné rovině jsou velmi rozměrné, ukázka přímo v programu je na Obr. 4.53. Jejich zpracování probíhá na řídicím počítači v reálném čase, a i přes jejich zřejmou komplexitu je výpočet

velmi rychlý a z pohledu časových konstant regulace trvá zanedbatelně dlouho.

4.7.6 Využití transformačních matic pro grafické vykreslování

```
// dolni koncovy bod sipky osy x
i = 5;
tmp1P.x = round(resolution.x/2 + resolution.x * ((
    plateRTCornerCalib.x + (float)(plateLTCornerCalib.x -
    plateRTCornerCalib.x) / 20 + (float)(plateRBCornerCalib.x + (
    float)(plateLBCornerCalib.x - plateRBCornerCalib.x) / 20 - (
    plateRTCornerCalib.x + (float)(plateLTCornerCalib.x -
    plateRTCornerCalib.x) / 20)) / 10 * (i + 0.15) - resolution.x
    /2) * px2mRatio.x + 0.3345*sin(gama) + (0.25 - 0.025)*(cos(beta
    )*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) - (0.25 - 0.025)*
    cos(gama) + (0.25 - (i + 0.15)*0.05)*sin(gama) - 0.3345*cos(
    alfa)*sin(gama) - (0.25 - (i + 0.15)*0.05)*cos(alfa)*sin(gama))
     / (2 * tanFOV2.x * (camPlateDist - (0.3345*sin(alfa) + (0.25 -
     (i + 0.15)*0.05)*sin(alfa) + (0.25 - 0.025)*cos(alfa)*sin(beta
    ))));

tmp1P.y = round(resolution.y/2 + resolution.y * ((
    plateRTCornerCalib.y + (float)(plateLTCornerCalib.y -
    plateRTCornerCalib.y) / 20 + (float)(plateRBCornerCalib.y + (
    float)(plateLBCornerCalib.y - plateRBCornerCalib.y) / 20 - (
    plateRTCornerCalib.y + (float)(plateLTCornerCalib.y -
    plateRTCornerCalib.y) / 20)) / 10 * (i + 0.15) - resolution.y
    /2) * px2mRatio.y + 0.3345*cos(gama) - (0.25 - 0.025)*(cos(beta
    )*sin(gama) - cos(gama)*sin(alfa)*sin(beta)) + (0.25 - (i +
    0.15)*0.05)*cos(gama) + (0.25 - 0.025)*sin(gama) - 0.3345*cos(
    alfa)*cos(gama) - (0.25 - (i + 0.15)*0.05)*cos(alfa)*cos(gama))
     / (2 * tanFOV2.y * (camPlateDist - (0.3345*sin(alfa) + (0.25 -
     (i + 0.15)*0.05)*sin(alfa) + (0.25 - 0.025)*cos(alfa)*sin(beta
    ))));
line(trackingFrame, Point(tmp2P.x, tmp2P.y), Point(tmp1P.x, tmp1P.
    y), Scalar(0, 0, 0), 2, LINE_AA);
```

Zdr. 27 Přepočet deformace dolního ramene šipky x-té osy při naklápění roviny

V podstatě na veškeré vykreslování do obrazu jsou aplikovány vztahy zahrnující odvozenou transformační matici, protože se vše promítá na plochu nakloněné

roviny. Postup pro zjištění výsledných pozic v obraze je obdobný, jako v kapitole 4.7.3. Vždy se musí vycházet z jedné známé pozice bodu určeného při kalibraci, od něj se vypočítá s pomocí transformační matice posun v GSS při náklonu roviny, a dosazením zjištěných parametrů do rovnice 4.8 se zahrne do přepočtu perspektivní zkreslení. Tímto postupem se zajistí, že se budou vykreslované objekty jevit jako promítnuté na nakloněnou rovinu.

K vykreslení není potřeba žádné kamerové detekce náklonu roviny ani algoritmů zpracování obrazu. Vše je čistě výpočetní záležitost, využívající kinematické transformační matice, rovnice odstraňující perspektivní zkreslení obrazu a výsledky kalibrační procedury, která byla provedena pouze jednou a jejíž výsledky byly uloženy do souboru, z kterého se při každém spuštění programu načítají. Reálné ověření správnosti výpočtů lze vidět na již zmiňovaném Obr. 4.45, nejlépe pozorovatelným prvkem je kružnice, která je z pohledu kamery deformovaná na elipsu, nebo osy souřadného systému systému nakloněné roviny, u nichž je přepočítávána i deformace ukazatele směru (šipky). Pro přepočet ukazatele směru osy je i následující ukázka ve Zdr. 27.

4.8 Řízení pohybu

V této podkapitole budou popsány použité metody řízení robotických struktur, jakožto hlavních představitelů nelineárních dynamických systémů s přesným řízením pohybu. Cílem je směřovat práci na metody nelineárního řízení, resp. řízení nelineárních objektů, jejichž nelinearita je uvažována už při návrhu regulátoru, příp. je rovnou zakomponována do zákona řízení [5, 63].

4.8.1 Matematický model

Matematický popis řízeného systému je nezbytný pro návrh strategie řízení. Měl by být co nejjednodušší, avšak musí stále s dostatečnou přesností vystihovat dynamiku řízeného systému. To znamená vždy kompromis mezi přesností a složitostí matematického modelu.

Pro odvození matematického popisu dynamiky řízeného systému budou použity Lagrangeovy pohybové rovnice 2. druhu v maticové formě. K tomu lze s výhodou využít kinematických transformačních matic a CAD modelu, bez kterých by byl přesný matematický popis takto komplexního systému v podstatě vyloučený. Ukázalo se, že i jednoduchý matematický model kopíruje s dostatečnou přesností dynamické chování celého systému, pokud je zjednodušení provedeno kvalifikovaně - viz autorova publikace [P. 4]. K tomuto zjednodušení je velice nápomocný dříve zmíněný CAD model; poskytuje např. přesnou hmotnost, polohu těžiště, popř. momenty/matice setrvačnosti, což jsou parametry zásadně ovlivňující vliv výsledného dynamického chování celé sestavy.

Model má celkem 6(4) zobecněných stupňů volnosti: 2-DOF pro náklon roviny, 2 DOF pro translaci kuličky po nakloněné rovině a 2 DOF pro její rotaci ve směru os roviny (ty se posléze vyjádří jako kombinace translace kuličky a jejího poloměru - rotace bez prokluzu - model se zjednoduší na 4-DOF). Obecně by systém měl 8-DOF, ale 6-DOF kuličky bylo omezeno na 2-DOF, protože v jedné ose zamezuje jejímu pohybu nakloněná rovina (pevnost materiálu) a gravitace (dotyk kuličky s nakloněnou rovinou - není uvažován odskok) a z 3-DOF pro rotaci kuličky okolo svých os je bráno v potaz pouze 2-DOF, a jen jako kombinace její translace. Kulička sice může ztratit kontakt s rovinou při rychlých změnách jejího náklonu, ale v tomto případě je stejně neřiditelná a matematický model má

sloužit primárně pro návrh zákona řízení. Uvažovat tedy o „přesnější“ variantě matematického popisu, je z praktického pohledu bezpředmětné.

Shrnutím výše uvedeného bude tedy systém popsán 4 diferenciálními rovnicemi 2. řádu se 4 zobecněnými DOF, které vyžadují určení kinetické a potenciální energie každé skupiny samostatně se pohybujících objektů. K určení kinetické energie je potřeba znát rychlosti jednotlivých částí v GSS, tedy vektory polohy těžišť ve zvolených LSS v závislosti na čase, a jejich hmoty (pro variantu zjednodušení na hmotné body, jinak i momenty/matice setrvačnosti). Pro určení potenciální energie je zase nutné zjistit polohu těžišť vůči ose gravitace a jejich hmotu. Pro nalezení všech poloh a rychlostí v závislosti na změně konfigurace modelu bude využita transformační matice, která byla odvozena v kapitole 4.7.4. K určení hmot/momentů/matic setrvačnosti a vzájemné polohy vycházející z kinematického uspořádání systému se zase využije CAD model (Obr. 4.4).

K odvození transformačních matic pro jednotlivé pohybové rovnice se bude vycházet z matice homogenní transformace v rovnici 4.12. Každé skupině vzájemně svázaných objektů musí náležet jedna transformační matice a bude označena podle toho, z jaké LSS provádí transformaci do GSS. Matice homogenní transformace byla odvozena obecně i s přihlédnutím na použití pro odvození matematického modelu, proto i značení souřadných systému stále koresponduje s Obr. 4.50.

$${}^{0}T_{\alpha} = {}^{0}T_{12\,(\beta=0;\ x=0;\ y=0;\ \varphi_X=0;\ \varphi_Y=0)}$$

$${}^{0}T_{\alpha} = \begin{bmatrix} \sin(\alpha)\sin(\gamma) & \cos(\gamma) & -\cos(\alpha)\sin(\gamma) & S_x + c\cdot\sin(\gamma) - c\cdot\cos(\alpha)\sin(\gamma) + bR\cdot\sin(\alpha)\sin(\gamma) \\ \sin(\alpha)\cos(\gamma) & -\sin(\gamma) & -\cos(\gamma)\cos(\alpha) & c\cdot\cos(\gamma) - S_y - c\cdot\cos(\alpha)\cos(\gamma) + bR\cdot\cos(\gamma)\sin(\alpha) \\ -\cos(\alpha) & 0 & -\sin(\alpha) & -c\cdot\sin(\alpha) - bR\cdot\cos(\alpha) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.16}$$

$${}^0T_\beta = {}^0T_{12\ (x=0;\ y=0;\ \varphi_X=0;\ \varphi_Y=0)}$$

$${}^0T_\beta = \begin{bmatrix} \cos(\beta)\sin(\alpha)\sin(\gamma) - \cos(\gamma)\sin(\beta) & \cos(\beta)\cos(\gamma) + \sin(\alpha)\sin(\beta)\sin(\gamma) & -\cos(\alpha)\sin(\gamma) & X_0 \\ \sin(\beta)\sin(\gamma) + \cos(\beta)\cos(\gamma)\sin(\alpha) & \cos(\gamma)\sin(\alpha)\sin(\beta) - \cos(\beta)\sin(\gamma) & -\cos(\alpha)\cos(\gamma) & Y_0 \\ -\cos(\alpha)\cos(\beta) & -\cos(\alpha)\sin(\beta) & -\sin\alpha & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.17}$$

$$X_0 = \ S_x + c\cdot\sin(\gamma) - bR\cdot(\cos(\gamma)\sin(\beta) - \cos(\beta)\sin(\alpha)\sin(\gamma)) - c\cdot\cos(\alpha)\sin(\gamma)$$

$$Y_0 = \ c\cdot\cos(\gamma) - S_y + bR\cdot(\sin(\beta)\sin(\gamma) + \cos(\beta)\cos(\gamma)\sin(\alpha)) - c\cdot\cos(\alpha)\cos(\gamma)$$

$$Z_0 = \ -c\cdot\sin(\alpha) - bR\cdot\cos(\alpha)\cos(\beta)$$

$${}^0T_x = {}^0T_{12\ (\alpha=0;\ y=0;\ \varphi_X=0;\ \varphi_Y=0)}$$

$${}^0T_x = \begin{bmatrix} -\cos(\gamma)\sin(\beta) & \cos(\beta)\cos(\gamma) & -\sin(\gamma) & S_x - bR\cdot\cos(\gamma)\sin(\beta) + x\cdot\cos(\beta)\cos(\gamma) \\ \sin(\beta)\sin(\gamma) & -\cos(\beta)\sin(\gamma) & -\cos(\gamma) & bR\cdot\sin(\beta)\sin(\gamma) - S_y - x\cdot\cos(\beta)\sin(\gamma) \\ -\cos(\beta) & -\sin(\beta) & 0 & -bR\cdot\cos(\beta) - x\cdot\sin(\beta) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.18}$$

$${}^0T_y = {}^0T_{12\ (\beta=0;\ x=0;\ \varphi_X=0;\ \varphi_Y=0)}$$

$${}^0T_y = \begin{bmatrix} \sin(\alpha)\sin(\gamma) & \cos(\gamma) & -\cos(\alpha)\sin(\gamma) & X_0 \\ \cos(\gamma)\sin(\alpha) & -\sin(\gamma) & -\cos(\alpha)\cos(\gamma) & Y_0 \\ -\cos(\alpha) & 0 & -\sin\alpha & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.19}$$

$$X_0 = \ S_x + c\cdot\sin(\gamma) - c\cdot\cos(\alpha)\sin(\gamma) + bR\cdot\sin(\alpha)\sin(\gamma) - y\cdot\cos(\alpha)\sin(\gamma)$$

$$Y_0 = \ c\cdot\cos(\gamma) - S_y - c\cdot\cos(\alpha)\cos(\gamma) + bR\cdot\cos(\gamma)\sin(\alpha) - y\cdot\cos(\alpha)\cos(\gamma)$$

$$Z_0 = \ -c\cdot\sin(\alpha) - bR\cdot\cos(\alpha) - y\cdot\sin(\alpha)$$

$${}^0T_{\varphi_X} = {}^0T_{12\ (\beta=0;\ x=0;\ y=0;\ \varphi_Y=0)}$$

$${}^0T_{\varphi_X} = \begin{bmatrix} r_{11} & \cos(\gamma) & \sin(\alpha)\sin(\varphi_X)\sin(\gamma) - \cos(\alpha)\cos(\varphi_X)\sin(\gamma) & X_0 \\ r_{21} & -\sin(\gamma) & \cos(\gamma)\sin(\alpha)\sin(\varphi_X) - \cos(\alpha)\cos(\varphi_X)\cos(\gamma) & Y_0 \\ r_{31} & 0 & -\cos(\alpha)\sin(\varphi_X) - \cos(\varphi_X)\sin(\alpha) & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.20}$$

$$r_{11} = \ \cos(\alpha)\sin(\varphi_X)\sin(\gamma) + \cos(\varphi_X)\sin(\alpha)\sin(\gamma)$$

$$r_{21} = \ \cos(\alpha)\cos(\gamma)\sin(\varphi_X) + \cos(\varphi_X)\cos(\gamma)\sin(\alpha)$$

$$r_{31} = \ \sin(\alpha)\sin(\varphi_X) - \cos(\alpha)\cos(\varphi_X)$$

$$X_0 = \quad S_x + c \cdot \sin(\gamma) - c \cdot \cos(\alpha)\sin(\gamma) + bR \cdot \sin(\alpha)\sin(\gamma)$$

$$Y_0 = \quad c \cdot \cos(\gamma) - S_y - c \cdot \cos(\alpha)\cos(\gamma) + bR \cdot \cos(\gamma)\sin(\alpha)$$

$$Z_0 = \quad -bR \cdot \cos(\alpha) - c \cdot \sin(\alpha)$$

$${}^0T_{\varphi Y} = {}^0T_{12\,(\alpha=0;\ x=0;\ y=0;\ \varphi_X=0)}$$

$${}^0T_{\varphi Y} = \begin{bmatrix} r_{11} & \cos(\beta)\cos(\varphi_Y)\cos(\gamma) + \cos(\gamma)\sin(\beta)\sin(\varphi_Y) & -\sin(\gamma) & S_x - bR \cdot \cos(\gamma)\sin(\beta) \\ r_{21} & -\cos(\beta)\cos(\varphi_Y)\sin(\gamma) - \sin(\beta)\sin(\varphi_Y)\sin(\gamma) & -\cos(\gamma) & bR \cdot \sin(\beta)\sin(\gamma)\, S_y \\ r_{31} & \cos(\beta)\sin(\varphi_Y) - \cos(\varphi_Y)\sin(\beta) & 0 & -bR \cdot \cos(\beta) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.21}$$

$$r_{11} = \quad \cos(\beta)\cos(\gamma)\sin(\varphi_Y) - \cos(\varphi_Y)\cos(\gamma)\sin(\beta)$$

$$r_{21} = \quad \cos(\varphi_Y)\sin(\beta)\sin(\gamma) - \cos(\beta)\sin(\varphi_Y)\sin(\gamma)$$

$$r_{31} = \quad -\sin(\beta)\sin(\varphi_Y) - \cos(\beta)\cos(\varphi_Y)$$

Vektory polohy těžišť pro jednotlivé LSS odvozených transformačních matic jsou obecně zapsány v rovnici 4.22.

$${}^{\alpha}r_{\alpha} = \begin{bmatrix} r_{\alpha x} \\ r_{\alpha y} \\ r_{\alpha z} \\ 1 \end{bmatrix}; \quad {}^{\beta}r_{\beta} = \begin{bmatrix} r_{\beta x} \\ r_{\beta y} \\ r_{\beta z} \\ 1 \end{bmatrix}; \quad {}^{x}r_{x} = {}^{y}r_{y} = {}^{\varphi_X}r_{\varphi_X} = {}^{\varphi_Y}r_{\varphi_Y} = \begin{bmatrix} r_{Bx} \\ r_{By} \\ r_{Bz} \\ 1 \end{bmatrix} \tag{4.22}$$

Matematický model - náhrada hmotnými body - postup 1

První postup k nalezení pohybových rovnic je založen na určení energií systému a využívá zjednodušení v podobě náhrady vzájemně se nepohybujících částí hmotnými body. Hmotný bod má nulové matice setrvačnosti, protože je veškerá hmota soustředěna v jednom bodě - těžišti každého článku. Záleží proto pouze na hmotnosti jednotlivých článků a na poloze jejich těžiště v konečné LSS (v související transformační matici). Výsledkem budou tedy 4 pohybové rovnice se 4-DOF, protože rotace kuličky není pro náhradu hmotným bodem uvažována.

Následující rovnice 4.23 - 4.26 slouží k určení kinetické a potenciální energie první pohyblivé části modelu, tedy části, kterou přímo ovlivňuje první DOF. Zbývající části jsou určeny obdobným způsobem s využitím odpovídajících transformačních matic a vektorů polohy těžišť.

Polohový vektor těžiště první pohyblivé části v GSS:

$$p_\alpha = {}^0T_\alpha \cdot {}^\alpha r_\alpha = \begin{bmatrix} p_{\alpha x} & p_{\alpha y} & p_{\alpha z} & 1 \end{bmatrix}^T \tag{4.23}$$

Vektor rychlosti těžiště první části v GSS:

$$v_\alpha = \frac{dp_\alpha}{dt} = \begin{bmatrix} v_{\alpha x} & v_{\alpha y} & v_{\alpha z} & 1 \end{bmatrix}^T \tag{4.24}$$

Kinetická energie první části:

$$Ek_\alpha = \tfrac{1}{2} \cdot m_1 \cdot \left(v_{\alpha x}^2 + v_{\alpha y}^2 + v_{\alpha z}^2\right) \tag{4.25}$$

A konečně potenciální energie první části (gravitace ve směru globální osy Z, proto $p_{\alpha z}$):

$$Ep_\alpha = m_1 \cdot g \cdot p_{\alpha z} \tag{4.26}$$

Po určení kinetické a potenciální energie všech zbylých pohyblivých částí modelu svázaných s jednotlivými DOF, je určena celková kinetická a potenciální energie systému a rovnou dosazena do Lagrangiánu:

$$\begin{aligned} Ek_{celk} &= Ek_\alpha + Ek_\beta + Ek_{Bx} + Ek_{By} \\ Ep_{celk} &= Ep_\alpha + Ep_\beta + Ep_{Bx} + Ep_{By} \\ L &= Ek_{celk} - Ep_{celk} \end{aligned} \tag{4.27}$$

Lagrangián je poté použit při odvození pohybových rovnic popisujících dynamické chování nakloněné roviny a kuličky:

$$\begin{array}{l} \frac{d}{dt}\left(\frac{\partial L}{\dot{\alpha}}\right) - \frac{\partial L}{\alpha} = Q_\alpha \\ \frac{d}{dt}\left(\frac{\partial L}{\dot{\beta}}\right) - \frac{\partial L}{\beta} = Q_\beta \\ \frac{d}{dt}\left(\frac{\partial L}{\dot{x}}\right) - \frac{\partial L}{x} = 0 \\ \frac{d}{dt}\left(\frac{\partial L}{\dot{y}}\right) - \frac{\partial L}{y} = 0 \end{array} \tag{4.28}$$

Q_α a Q_β jsou točivé momenty motorů pro první a druhý DOF. Rovnice 4.28 může být pro lepší přehlednost přepsána do maticové formy pohybových rovnic (4.29), kde je fyzikální význam jednotlivých částí více patrný:

$$\mathrm{D}(\bar{\mathbf{q}}) \cdot \ddot{\bar{\mathbf{q}}} + \bar{\mathbf{H}}\left(\bar{\mathbf{q}}, \dot{\bar{\mathbf{q}}}\right) + \bar{\mathbf{G}}\left(\bar{\mathbf{q}}\right) = \bar{\mathbf{Q}}$$

$$D\left(\alpha, \beta, x, y\right) \cdot \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{x} \\ \ddot{y} \end{bmatrix} + \bar{H}\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}, x, \dot{x}, y, \dot{y}\right) + \bar{G}\left(\alpha, \beta, x, y\right) = \begin{bmatrix} Q_\alpha \\ Q_\beta \\ 0 \\ 0 \end{bmatrix} \tag{4.29}$$

kde:

$\mathrm{D}(\bar{\mathbf{q}})$ ... symetrická matice setrvačnosti
$\bar{\mathbf{H}}\left(\bar{\mathbf{q}}, \dot{\bar{\mathbf{q}}}\right)$ ... vektor vazeb rychlosti
$\bar{\mathbf{G}}\left(\bar{\mathbf{q}}\right)$ ... vektor gravitační síly
$\bar{\mathbf{Q}}$ ... nepotenciálová (nekonzervativní) zobecněná síla způsobující změnu $\bar{\mathbf{q}}$

Pro zobecnění konkrétního postupu uvedeného v rovnicích 4.23 - 4.28 se dají jednotlivé členy rovnice 4.29 rozepsat do tvaru:

$$\mathbf{D_{i,j}} = \sum_{r=max\{i,j\}}^{n} tr\left[\frac{\partial\, {}^0T_r}{\partial q_j} \cdot {}^r\bar{I}_r \left(\frac{\partial\, {}^0T_r}{\partial q_i}\right)^T\right] \tag{4.30}$$

$$\mathbf{H_i} = \sum_{k=1}^{n}\sum_{m=1}^{n} H_{ikm}\, \dot{q_k}\, \dot{q_m} \tag{4.31}$$

$$\mathbf{H_{ikm}} = \sum_{r=max\{i,k,m\}}^{n} tr\left[\frac{\partial^2 {}^0T_r}{\partial q_k \, \partial q_m} \cdot {}^r\bar{I}_r \left(\frac{\partial {}^0T_r}{\partial q_i}\right)^T\right] \tag{4.32}$$

$$\mathbf{G_i} = -\sum_{r=i}^{n} m_r \left({}^0\bar{g}\right)^T \cdot \frac{\partial {}^0T_r}{\partial q_i} \cdot {}^r\bar{r}_r \tag{4.33}$$

kde:

n ... počet článků (DOF)
i, j ... stavové proměnné
$\bar{I}$ ... matice pseudosetrvačnosti - obecná definice v rovnici 4.34
${}^0\bar{g}$ ... vektor gravitace v GSS
${}^r\bar{r}_r$ ... polohový vektor těžiště daného článku v LSS

$$\bar{\mathbf{I}} = \begin{bmatrix} \frac{-I_{xx}+I_{yy}+I_{zz}}{2} & I_{xy} & I_{xz} & mr_x \\ I_{yx} & \frac{I_{xx}-I_{yy}+I_{zz}}{2} & I_{yz} & mr_y \\ I_{zx} & I_{zy} & \frac{I_{xx}+I_{yy}-I_{zz}}{2} & mr_z \\ mr_x & mr_y & mr_z & m \end{bmatrix} \tag{4.34}$$

Rovnice 4.30 - 4.34 jsou při zjištění čtyř klíčových parametrů algoritmizovatelné, je tedy možné je zapsat do skriptu pro matematický software, který zajistí plně automatizovaný výpočet této komplikované soustavy rovnic. Tím se výrazně sníží nejen pracnost s výpočtem, ale především se obejdou potenciální místa, kde mohou vzniknout chyby. Zmiňovanými parametry nutnými pro algoritmizování problému jsou: **matice homogenní transformace** (kinematická transformační matice z D-H notace), **polohový vektor těžiště** v poslední LSS transformační matice, **hmotnost článku** a **matice setrvačnosti** článku (pro případ zjednodušení na hmotný bod bude matice setrvačnosti nulová). Pro přesné zjištění všech uvedených parametrů je klíčovým prvkem CAD model, protože minimálně určení polohového vektoru těžiště je pouhým odhadem/měřením u takto komplexního modelu velice komplikované, je-li vůbec možné, a má zásadní vliv na

přesnost celého matematického modelu.

Při využití maticové formy pohybových rovnic a jejich algoritmizaci je už na uživateli, jestli zvolí jejich odvození se zjednodušením na hmotné body, nebo bude využívat přístup přes matice setrvačnosti a tím dostane ten nejpřesnější matematický model, který je vůbec možné odvodit. Pracnost jednoho i druhého postupu je stejná, stačí pouze ponechat, nebo správně eliminovat matici setrvačnosti, a spustit skript. MATLAB skripty pro oba přístupy jsou uvedeny v PŘÍLOHÁCH A, B a C.

PŘÍLOHA A obsahuje postup výpočtu uvedený v rovnicích 4.23 - 4.28. Tento postup je sestavený přesně na probíraný systém, je pracný a při jeho modifikaci velmi lehce vznikne chyba, která je těžko zjistitelná (správnost odvozeného modelu prozatím není s čím porovnat).

Matematický model - originál bez zjednodušení - postup 2

PŘÍLOHA B obsahuje zmíněný automatizovaný výpočet pro 6 pohybových rovnic se 4 zobecněnými DOF, který je použitý pro odvození originálního nezjednodušeného matematického modelu - jsou použity matice setrvačnosti a je uvažován i vliv rotace kuličky. Výsledné rovnice jsou v příloze také zmíněny a jsou dále upravovány do takové míry, aby se dynamické chování původního a zjednodušeného systému co nejméně lišilo a zároveň byly rovnice co nejjednodušší. Výsledkem je vždy kompromis, ale pro použití matematického modelu k návrhu zákona řízení je zjednodušení nutné - původní rovnice jsou pro převod do stavového prostoru příliš komplikované a vylučují další úpravy. Model také předpokládá již dříve zmíněné dokonalé odvalování kuličky po nakloněné rovině (bez prokluzu), protože snímání úhlového natočení kuličky kolem všech jejich os rotace je velmi komplikované (a v tomto řešení nerealizované), dochází k němu jen velmi zřídka a na velmi krátkou dobu (v kritických fázích s vysokou dynamikou) a musely by být brány v potaz i změny v matematickém popisu řízeného děje. Šest pohybových rovnic s přepočtem 5. a 6. DOF na 3. a 4. DOF je odvozeno z důvodu kontroly správnosti výpočtu, na základě porovnání výsledků s fyzikálním modelem, a také pro zjednodušení původních pohybových rovnic do formy vhodné pro další syntézu. Ve fyzikálním modelu totiž nelze přímo snímat kombinované silové účinky kuličky (kombinace silových účinků od translace a

rotace), ale pouze výslednici translační síly kuličky a točivého momentu kuličky vůči zvolené souřadné soustavě (porovnání v kapitole 4.8.2).

PŘÍLOHA C už obsahuje odvození finálních 4 pohybových rovnic se 4-DOF, kde jsou silové účinky kuličky od translace i rotace sloučeny do jedné rovnice pro každou osu (2 DOF). Tyto rovnice jsou opět zjednodušeny do míry odpovídající aktuálním požadavkům a následně použity pro odvození stavového popisu - rovnice 4.38. Konkrétní parametry modelu, ať už konstrukční, tak materiálové, jsou k dispozici taktéž v PŘÍLOHÁCH A až C. Původní pohybová rovnice byla výrazně zredukována (cca 10x) na výsledný tvar v rovnici 4.35. Toto zjednodušení má stále zachovanou původní dynamiku přechodového děje a oproti originálu se liší minimálně (Obr. 4.54 + PŘÍLOHA D a E).

$$\begin{aligned} Q_\alpha &= 0.2716 \cdot \ddot{\alpha} + (0.4712 + 0.3345 \cdot m_3 + m_3 \cdot y) \cdot g \cdot \cos(\alpha) \\ Q_\beta &= 0.05051 \cdot \ddot{\beta} + g \cdot m_3 \cdot x \cdot \cos(\beta) \\ 0 &= \tfrac{7}{5} \cdot m_3 \cdot \left(\ddot{x} - bR \cdot \ddot{\beta}\right) + g \cdot m_3 \cdot \sin(\beta) \\ 0 &= \tfrac{7}{5} \cdot m_3 \cdot (\ddot{y} - bR \cdot \ddot{\alpha}) - 0.3345 \cdot m_3 \cdot \dot{\alpha}^2 + g \cdot m_3 \cdot \sin(\alpha) \end{aligned} \tag{4.35}$$

Po přepisu rovnice 4.35 do maticové formy pohybových rovnic budou jednotlivé matice vypadat následovně:

$$D = \begin{bmatrix} 0.2716 & 0 & 0 & 0 \\ 0 & 0.05051 & 0 & 0 \\ 0 & -\frac{7}{5} \cdot bR \cdot m_3 & \frac{7}{5} \cdot m_3 & 0 \\ -\frac{7}{5} \cdot bR \cdot m_3 & 0 & 0 & \frac{7}{5} \cdot m_3 \end{bmatrix} \tag{4.36}$$

$$H = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -0.3345 \cdot m_3 \cdot \dot{\alpha}^2 \end{bmatrix}; \quad G = \begin{bmatrix} (0.4712 + 0.3345 \cdot m_3 + m_3 \cdot y) \cdot g \cdot \cos(\alpha) \\ g \cdot m_3 \cdot x \cdot \cos(\beta) \\ g \cdot m_3 \cdot \sin(\beta) \\ g \cdot m_3 \cdot \sin(\alpha) \end{bmatrix}$$

Jak je z rovnice 4.36 patrné, hlavní prvky ovlivňující dynamiku systému jsou vázané na setrvačné hmoty modelu (matice D) a gravitační sílu (matice G). Od-

středivé/dostředivé síly obsažené v matici H se objevují ve zjednodušené formě pohybových rovnic pouze u 4. DOF, Coriolisovy síly nejsou přítomny vůbec. V originálním matematickém modelu pochopitelně jsou, ale pro požadavky dynamického chování tohoto systému je jejich vliv zanedbatelný. Stejně je tomu u mnoha dalších složek, kterými byly původně matice D, H a G naplněny, a později kvalifikovaně vyloučeny.

Pokud jsou jednotlivé pohybové rovnice rozebrány detailněji, je možno bez ohledu na to, jak systém reálně vypadá říci, že např.:

- Silové účinky motoru v prvním DOF (α), ovlivňují motor v druhém DOF pouze zanedbatelně.
- Silové účinky motoru v druhém DOF (β), ovlivňují motor v prvním DOF pouze zanedbatelně.
- Zásadní vliv na momentové požadavky obou motorů, má kromě hmotností pohyblivých částí sestavy, také poloha kuličky - ale pouze v odpovídajících osách (y pro α, x pro β).
- Rychlost a zrychlení kuličky i roviny v druhém DOF ovlivňují točivý moment motoru v prvním DOF pouze zanedbatelně.
- Rychlost a zrychlení kuličky i roviny v prvním DOF ovlivňují točivý moment motoru v druhém DOF pouze zanedbatelně.
- Jediná významná odstředivá síla v sestavě je od rychlosti změny prvního DOF (α), a ovlivňuje pouze pohyb kuličky v ose y.
- Hmotnost kuličky má nezanedbatelný vliv nejen na momentové požadavky aktuátorů, ale také na celkové chování modelu.
- Jeden aktuátor řídí dva DOF - 1. DOF je přes gravitační sílu spjatý se 4. DOF, 2. DOF přes gravitační sílu zase se 3. DOF.
- Dosazením konkrétních hodnot do stavových proměnných v matici G je zjištěný potřebný moment motorů pro udržení nakloněné roviny ve zvolené konfiguraci (v ustáleném stavu).

Z pohybových rovnic v 4.36 byly dále pro účely stavového zápisu vyjádřeny druhé derivace stavových proměnných. Aby bylo možné stavové proměnné určit s využitím softwarových prostředků (např. MATLAB), byly přepsány do následující podoby:

$$D \cdot \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{x} \\ \ddot{y} \end{bmatrix} = Q - H - G \qquad \Rightarrow \qquad \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{x} \\ \ddot{y} \end{bmatrix} = D^{-1} \cdot (Q - H - G) \tag{4.37}$$

Konkrétní postup je uvedený v PŘÍLOZE B a C. Výsledkem jsou stavově zapsané pohybové rovnice uvedené v rovnici 4.38.

$$\begin{aligned}
\ddot{\alpha} &= 3.682 \cdot [Q_\alpha - (0.4712 + 0.3345 \cdot m_3 + m_3 \cdot y) \cdot g \cdot \cos(\alpha)] \\
\ddot{\beta} &= 19.8 \cdot [Q_\beta - g \cdot m_3 \cdot x \cdot \cos(\beta)] \\
\ddot{x} &= 19.8 \cdot bR \cdot [Q_\beta - g \cdot m_3 \cdot x \cdot \cos(\beta)] - \frac{5}{7} \cdot g \cdot \sin(\beta) \\
\ddot{y} &= 3.682 \cdot bR \cdot [Q_\alpha - (0.4712 + 0.3345 \cdot m_3 + m_3 \cdot y) \cdot g \cdot \cos(\alpha)] + \\
&\quad + \frac{5}{7} \cdot \left[0.3345 \cdot \dot{\alpha}^2 - g \cdot \sin(\alpha)\right]
\end{aligned} \tag{4.38}$$

Rovnice byly upraveny do takového tvaru, aby bylo jasně viditelné spojení mezi 1. a 4. DOF ($\ddot{\alpha}$ a $\ddot{y}$) a 2. a 3. DOF ($\ddot{\beta}$ a $\ddot{x}$). Ve stavovém zápisu už jsou rovnice velmi dobře čitelné a lze z nich např. vyčíst, že:

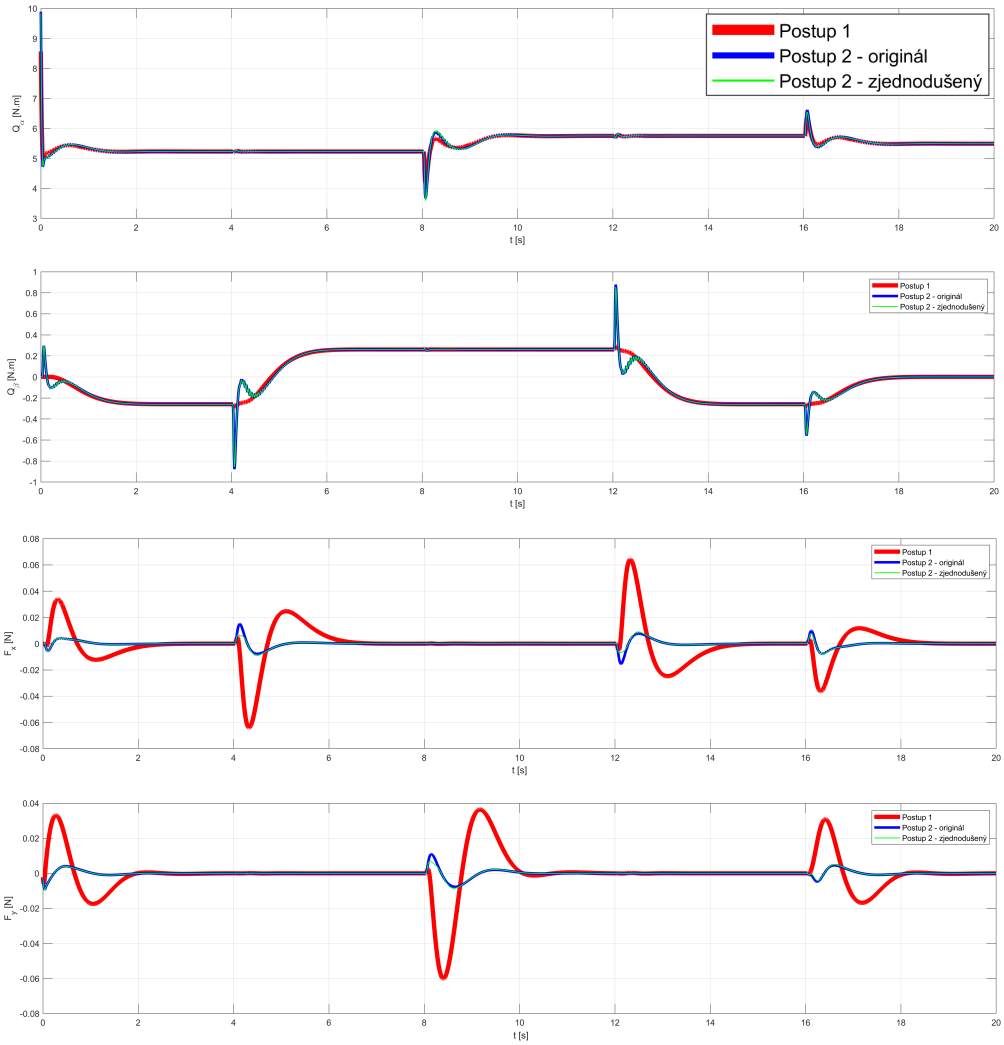
- Změna stavových veličin spojených s α a y neovlivňuje chování soustavy v 2. a 3. DOF (změnu souřadnic v β a x). To platí i obráceně.
- Hmotnost kuličky má významný vliv na všechny stavové proměnné.
- Poloměr kuličky, související s jejím momentem setrvačnosti, má vliv pouze na 3. a 4. DOF (dynamika pohybu kuličky po nakloněné rovině).
- Kladný posun kuličky v ose y je spjatý s nárůstem točivého momentu motoru pro 1. DOF - v případě udržení rovnovážného stavu.
- Kladný posun kuličky v ose x je spjatý s nárůstem točivého momentu motoru pro 2. DOF - v případě udržení rovnovážného stavu.
- Hmota roviny je uložena symetricky vůči ose rotace roviny pro β (vyváženě), a taktéž nulová pozice kuličky v ose x leží na ose rotace pro úhel β.
- Hmota roviny je uložena nesymetricky vůči ose rotace roviny pro α (nevyváženě) a taktéž nulová pozice kuličky v ose y neleží na ose rotace pro

úhel α - od osy rotace je vzdálena 33,45 cm.

- Pokud je zanedbán vliv zrychlení 2. DOF na změnu souřadnice kuličky v ose x, bude se kulička při kladném úhlu náklonu β pohybovat v záporném směru x.
- Pokud je zanedbán vliv zrychlení a odstředivé síly od 1. DOF na změnu souřadnice kuličky v ose y, bude se kulička při kladném úhlu náklonu α pohybovat v záporném směru y.

Opět je ale nutné zdůraznit, že stavový zápis je vytvořený ze zjednodušeného matematického modelu, je tedy taky zjednodušený. Různé přístupy k odvození matematických modelů byly již dříve slovně porovnávány z pohledu vzájemné podobnosti dynamických průběhů. Ony dynamické průběhy, na základě kterých byly matematické modely vzájemně kontrolovány a které také sloužily v podstatě jako jediná zpětná vazba k určení vhodné míry zjednodušení původního matematického modelu, jsou uvedeny na grafech v Obr. 4.54.
Ostatní testované průběhy jsou uvedeny v PŘÍLOZE D (testovací průběh pro typovou trajektorii kuličky ve tvaru kružnice - ustálený stav s konstantními rychlostmi kuličky a harmonickými úhlovými rychlostmi náklonu roviny) a PŘÍLOZE E (testovací průběh pro typovou trajektorii kuličky ve tvaru asteroidy - maximální dynamika změny všech stavových parametrů). Na Obr. 4.54, stejně jako v PŘÍLOHÁCH D a E, jsou zobrazeny grafické průběhy točivých momentů obou motorů (Q_α a Q_β) a silových účinků kuličky v ose x a y (F_x a F_y), při typové trajektorii s maximální dynamikou pohybu, která bude po systému vyžadována. Jsou zde vykresleny celkem 3 trajektorie:

- Průběh s **červenou** barvou je získán z matematického modelu odvozeného dle prvního postupu - náhrada pohyblivých částí hmotnými body - rovnice 4.23 - 4.28.
- Průběh s **modrou** barvou náleží výsledkům získaných z matematického modelu odvozeného podle druhého postupu - každé pohyblivé části odpovídá matice setrvačnosti, která obsahuje informaci o rozložení hmoty - originální matematický model bez zjednodušení - 4.30 - 4.34.
- Průběh se **zelenou** barvou pochází ze zjednodušeného tvaru originálního matematického modelu - rovnice 4.35.



Obr. 4.54 Porovnání grafických průběhů pro typovou trajektorii kuličky ve tvaru čtverce

Z průběhů (i v PŘÍLOHÁCH D a E) je patrné, že křivka pro zjednodušený matematický model, se až na zanedbatelné nuance shoduje s křivkou originálního matematického modelu. To už však neplatí pro průběhy odpovídajícím náhradě pohyblivých částí hmotnými body. U křivky točivého momentu druhého motoru Q_β je neshoda v přechodových dějích - neobsahují žádné momentové špičky

a dynamika je tedy značně zkreslená. Složitost výsledné rovnice pro zjednodušení hmotnými body se přesto blíží složitosti zápisu originálního matematického modelu, je tedy velice komplikovaná, a v přesnosti nedosahuje ani výrazně zjednodušené rovnici 4.35. Je to způsobeno pozicí těžiště druhé pohyblivé části na ose rotace β a z podstaty věci je zanedbán vliv velikosti nakloněné roviny v ose x. Jinými slovy, při tomto zjednodušení nezáleží na velikosti nakloněné roviny. Pokud bude mít stejnou hmotnost, její dynamické projevy jsou totožné - a to pochopitelně není pravda, z pohledu dynamického chování má velikost roviny zásadní vliv na její dynamické vlastnosti.

U křivek pro silové účinky kuličky v ose x a y (F_x a F_y), se o podobnosti už ani mluvit nedá. Průběhy pro postup 1 jsou přesto správné, ale pouze pokud je uvažována složka translace a zanedbána rotace kuličky, což vychází z podstaty věci při náhradě hmotnými body. Že je tomu opravdu tak, je možné ověřit na průbězích, kde je porovnáván matematický model s 6-DOF s modelem fyzikálním na Obr. 4.65 a PŘÍLOHÁCH G a H. Matematický popis se zjednodušením na hmotné body tedy není vhodný pro tyto účely a bude již dále vyřazen z oblasti zájmu.

Jak bylo ale ověřeno, že jsou matematické modely odvozeny správně a že během výpočtu nevznikla žádná chyba? Kontrola správnosti mezikroků je také obtížná, protože výsledný tvar modelu pochopitelně není známý. Je zde mnoho míst, kde se lze zmýlit, aniž by si toho kdokoliv všiml, i pokud probíhá výpočet pohybových rovnic automatizovaně a jeho algoritmus je zapsaný správně. Detekce chyby je potom velice komplikovaná a může vzniknout např. při:

- Nesprávně zakreslených souřadnicových systémech ve fázi odvozování transformačních matic.
- Nedodržení některé z podmínek v D-H notaci a obecně tedy nesprávné transformační matici (byť jediný řádek/parametr v jedné z transformačních matic) - při kontrole kinematiky se navíc vůbec nemusí projevit.
- Otočení směru (znaménka) některé stavové proměnné oproti reálnému systému.
- Chybný přepis/kopírování parametrů z CAD modelu (hmota, vektory, matice setrvačnosti).

- Nesouhlasné umístění/orientace vztažného souřadnicového systému v CAD modelu vůči D-H notaci.
- Nesprávně zvolený vztažný souřadnicový systém při zjišťování vektoru těžiště/matice setrvačnosti.
- Nekorespondující jednotkový systém v CAD modelu oproti simulaci/reálnému modelu.
- Chybně nastavené materiály u některých částí v CAD modelu.
- Přehlédnutí některé z mnoha částí sestavy, při zjišťování fyzikálních vlastností v CAD modelu.
- Chybný zápis při přepisu obecných vztahů do algoritmu výpočtu.
- Obecně neznalost propojení obecných matematických přístupů k vyřešení konkrétního technického problému.

To byly pouze některé z hlavních oblastí, v kterých je jednoduché se chyby dopustit, a u kterých by měla být zvýšená pozornost na to, co konkrétně je potřeba zjistit a následně dosadit. Stále však nelze říci, že je výsledná pohybová rovnice odvozena správně. Dosazením konkrétních hodnot v ustálených stavech je možné pouze odhadnout, jestli by výsledek mohl být správně, nebo je naprosto nereálný. Ověřit správnost odvození dynamické části rovnice při nenulových derivacích stavových veličin je ale pouhým odhadem vyloučené.

Ani při dostupnosti snímaných dat z reálného modelu není tato komplikace moc usnadněna, protože na jednu stranu je reálný model zatížen poruchami, které do matematického modelu buď není možné zahrnout, nebo je problematické je matematicky popsat, na druhou stranu nejsou některé parametry ani měřitelné (např. sílové účinky kuličky v jednotlivých osách zvoleného souřadného systému). Ve výsledku tedy stejně nelze ověřit, jestli a čím jsou způsobeny rozdíly mezi matematickým modelem a reálně naměřenými daty. Jedním z potenciálně vhodných řešení by bylo mít k dispozici jiný podobně orientovaný, matematický/fyzikálně popsaný model, který není založený na výše uvedeném postupu odvozování, není tedy ovlivněn možnou chybou uživatele a lze o něm tvrdit, že je s vysokou mírou pravděpodobnosti správný.

Tento model lze s využitím správně sestaveného CAD modelu vytvořit, je nazýván fyzikálním a je popsán v další podkapitole.

4.8.2 Fyzikální model

Testy kvality výsledné regulace se neprovádějí na matematickém modelu, ale na modelu fyzikálním, který je považován za nejlepší možnou náhradu reálného systému. Jeho přesnost (podobnost s reálným systémem) se odvíjí od CAD modelu, na kterém je založený - především na správné volbě rozměrů, materiálů a kinematických vazeb.

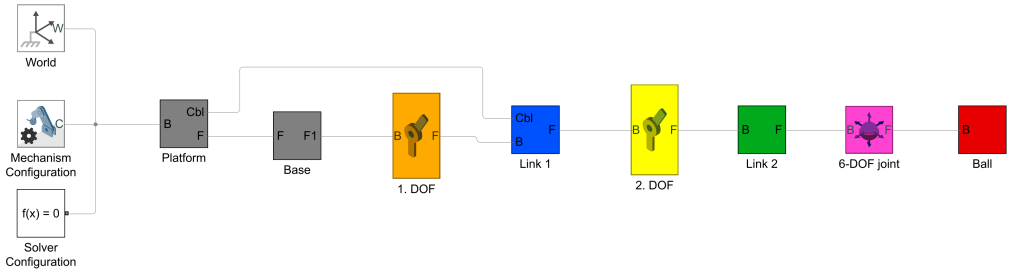
Dalo by se namítnout, že i za fyzikálním modelem se stále skrývá matematická interpretace. Je tomu samozřejmě tak, rozdíl je zde ovšem zásadní - ve fyzikálním modelu se za každým objektem skrývá různě komplikovaná/komplikované diferenciální rovnice (podle charakteru a nastavení každého objektu), naproti tomu matematický model má několik diferenciálních rovnic a v těchto rovnicích jsou zahrnuty všechny vlivy najednou, neoddělitelně. Stejně detailní matematický model jako model fyzikální by byl extrémně složitý a v podstatě nepoužitelný na odvození zákona řízení - v drtivé většině případů by ani nešel převést do stavového prostoru (alespoň ne do tvaru použitelného k další analýze a syntéze). Z fyzikálního modelu, který je zde popisován, ani není možné zpětně zjistit matematický zápis, není tedy k dispozici žádný základ pro návrh zákona řízení. Kvůli komplexitě systému selhávají i metody induktivní identifikace.

Jak bylo v předchozí podkapitole uvedeno, už v základu byl matematický model zjednodušený z reálných 8-DOF na 4-DOF. Přesněji řečeno, 6-DOF kuličky bylo zjednodušeno na zobecněné 2-DOF - důvody a popis tohoto zjednodušení jsou taktéž uvedeny v podkapitole popisující matematický model. Fyzikální model ale žádné z těchto zjednodušení neobsahuje, má plných 8-DOF, a i kdyby tedy byl matematický model odvozený správně a ponechaný bez zjednodušení, fyzikální model bude vždy detailnější a přesnější. Obsahuje navíc i nastavitelné tření v kloubech, silové dorazy, pružnost vazeb, vůle apod. Z uvedeného je patrné, že pro správný a vůbec možný návrh regulace a následné testování spolehlivosti a robustnosti je výhodné mít k dispozici jak model matematický, tak fyzikální [57, 18].

Pokud je k dispozici CAD model, není vytvoření základního fyzikálního modelu nic komplikovaného. Nejdříve je potřeba zkontrolovat, jestli je každá použitá část správně vyvazbena a má nastavený odpovídající materiál. Pokud je

CAD model některé části zjednodušený, je potřeba nastavit jeho materiálové vlastnosti ručně - podle významnosti daného dílu. Jinými slovy, nevadí, pokud nejsou přenastaveny vlastnosti plastového distančního sloupu, který byl vytištěný s částečnou výplní a je namodelován s výplní plnou, protože jeho vliv na dynamické projevy modelu jako celku je zanedbatelný. Pokud je ale opomenuto, že je model motoru v druhém DOF namodelován jako plný ocelový kvádr a jsou mu ponechány jeho vlastnosti vyplývající z nastaveného materiálu, je to už významný zásah do dynamiky modelu, který není zanedbatelný - už z pouhého porovnání hmotností reálného motoru a motoru v CAD modelu.

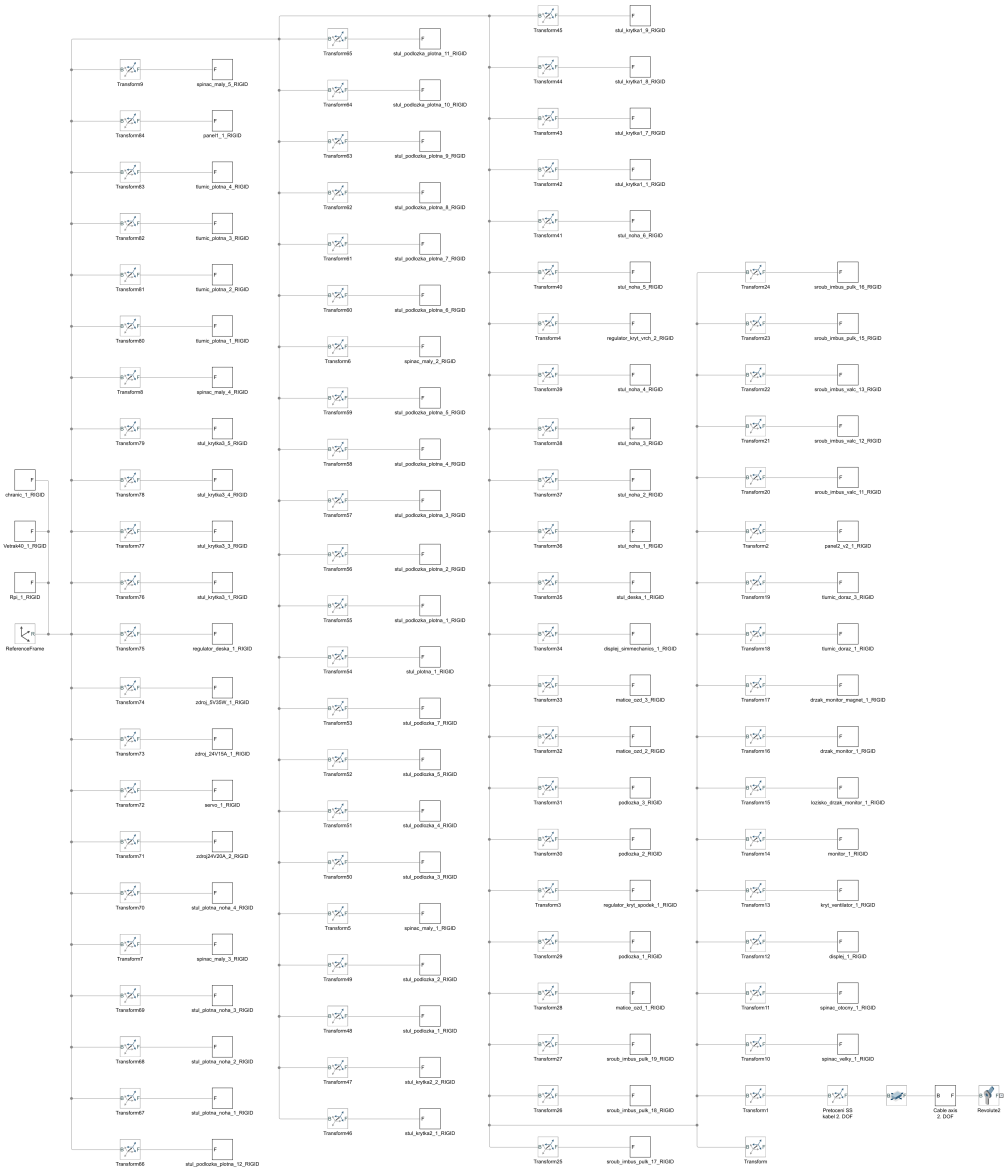
Po nastavení CAD modelu se provede export přímo z prostředí *SolidWorks* do *Simscape Multibody*. Tento export vytvoří rozsáhlý *.xml* soubor (podle složitosti CAD modelu) s informacemi o provázanosti a materiálových vlastnostech taktéž exportovaných *.step* modelů všech částí sestavy. Aby byl export možný, je nutné stáhnout a do *MATLABu* nainstalovat zásuvný modul umožňující toto propojení. V SolidWorks se poté objeví záložka pro export. Nakonec už se jen v *MATLABu* tento *.xml* soubor importuje a pokud jsou MATLABem podporovány všechny vazby nastavené v *SolidWorks*, vytvoří se kompletní fyzikální model. Z pohledu vazeb jsou ovšem podporovány jen základní vazby a např. omezení 6-DOF vazby kuličky je nutné napsat ručně. Základní importovaný Simscape model je uvedený na Obr. 4.55.



Obr. 4.55 Základní importovaný fyzikální model v Simscape Multibody

Model je na první pohled velice jednoduchý, ale obsahuje celkově přes 270 dílů, tedy *.step* modelů, které jsou uzavřeny v jednotlivých subsystémech, jako je *Platform*, *Base*, *Link 1*, *Link 2* a *Ball*. Tyto subsystémy obsahují všechny díly s kinematickými vazbami, které jsou vzájemně nepohyblivé, tedy tvoří pevnou

vazbu. Jsou zde pouze pro zobrazovací účely, jejich materiálové vlastnosti se načítají ze souboru vygenerovaného při importu a mohly by být nahrazené jedním dílem, který by obsahoval souhrnné informace.



Obr. 4.56 Rozbalený subsystém *Platform* představující díly (*.step* modely) základny s vazbami

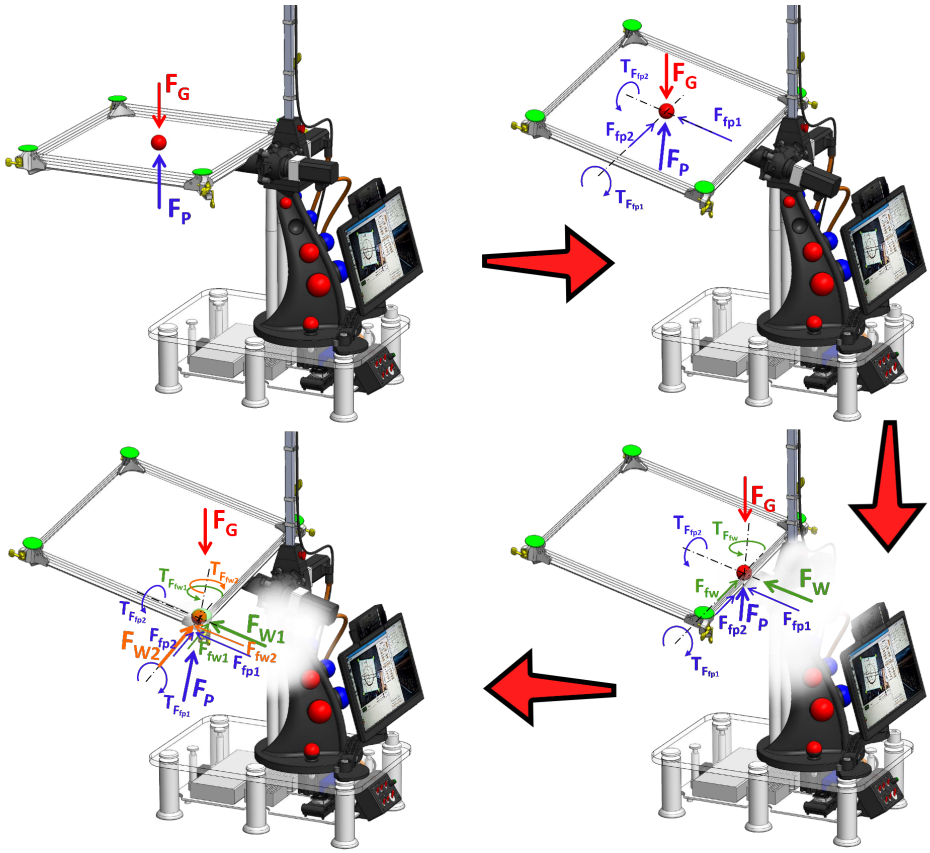
Pro příklad je na Obr. 4.56 rozbalený subsystém *Platform*, který představuje díly (*.step* modely) základny.

Do fyzikálního modelu na Obr. 4.55 bylo pro jeho základní funkčnost nutné přidat omezení 6-DOF vazby kuličky a silové dorazy, aby nedocházelo k prolínání objemů. Podle pozice, rychlosti a zrychlení kuličky na nakloněné rovině budou na kuličku působit různé síly a momenty, které je nutné matematicky zapsat do skriptu. Bude se jednat o:

- Direkční sílu zabraňující kuličce propadnout nakloněnou rovinou.
- Sílu tření mezi kuličkou a podložkou, aby bylo zajištěno její odvalování.
- Sílu tření mezi kuličkou a zábranami, ohraničujících nakloněnou roviny, aby náraz do nich ovlivňoval její pohyb.
- Direkční sílu od zábran, která způsobí odražení kuličky podle jejího úhlu dopadu a nárazové rychlosti.
- Točivé momenty působící na kuličku, jako reakce na třecí sílu, způsobující její odvalování.
- Omezení direkční síly od podložky pouze na rozměry nakloněné roviny - pokud se kulička dostane mimo podložku, spadne z ní.
- Omezení direkční síly zábran pouze do jejich výšky - kulička může při dynamickém pohybu roviny zábrany přeskočit.
- Omezení maximální direkční síly zábran vyplývajících z jejich materiálových vlastností - kulička prorazí zábrany a spadne z podložky, pokud je nárazové energie větší než je pevnost zábran.

Zápis v MATLAB skriptu je uvedený v PŘÍLOZE F. Rozbor různých situací pro stěžejní části popisu fyzikální reality je souhrnně demonstrován na Obr. 4.57. Jsou zde rozebrány 4 hlavní situace, ke kterým při pohybu dochází:

- Rovina je ve vodorovné poloze, rychlost kuličky je nulová.
- Rovina je nakloněná tak, že má kulička tendenci směřovat k levému dolnímu rohu nakloněné roviny (z pohledu uživatele stojícího před monitorem modelu).
- Náklon roviny je stejný jako dříve, kulička už narazila na zábranu ohraničení a směřuje s 2 body dotyku (podložka a zábrana) do levého dolního

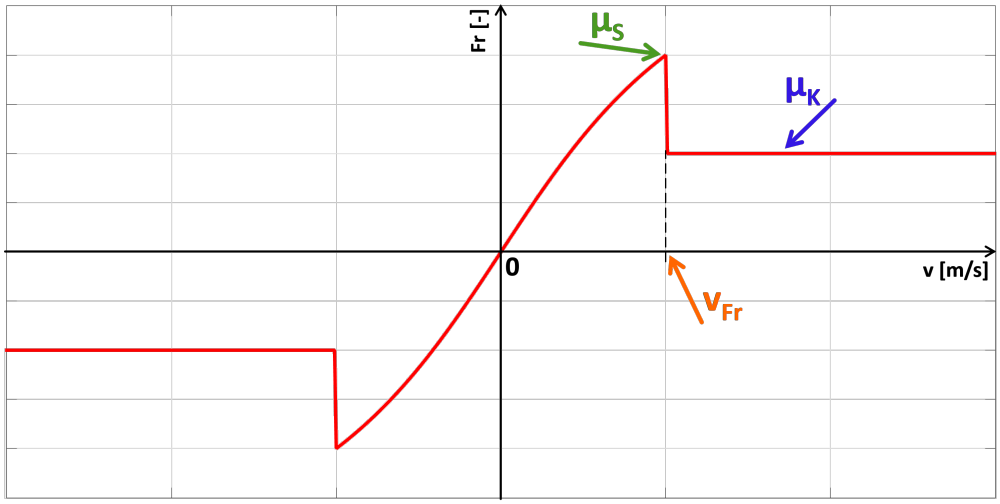


Obr. 4.57 Různé situace sil a momentů působících na kuličku během pohybu

rohu roviny.

- Náklon roviny je stejný jako dříve, předchozí situace se rozšířila o náraz kuličky do další zábrany ohraničení (3 body dotyku - podložka a 2 zábrany).

Použitá charakteristika suchého tření je zobrazena na Obr. 4.58. Vztah určující celkové kombinované tření použitý v simulaci je uveden v rovnici 4.39. Ten říká, že když je rychlost skluzu $v_{Slip}[\omega(t), v(t)]$ nižší, než definovaná maximální rychlost skluzu v_{Fr}, je tření ve své statické části s maximální hodnotou μ_S. Když však rychlost skluzu překročí v_{Fr}, tření se sníží na kinematickou konstantu tření μ_K. Výsledná třecí síla F_{frict} je založena aktuální hodnotě suchého tření v charakteristice na Obr. 4.58, vynásobené normálovou silou F_N, kterou kulička působí na podložku/zábrany. Zahrnuto je také lineární tření μ_{Lin}, které se přičítá k



Obr. 4.58 Zjednodušená charakteristika statického a kinematického tření použitá v simulaci

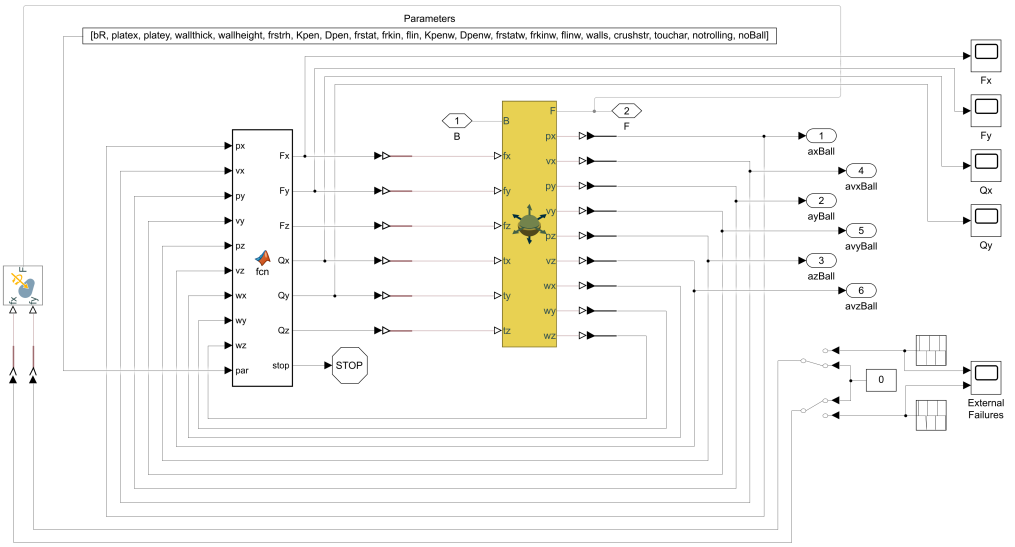
výsledné třecí síle a závisí na rychlosti posuvu kuličky $v(t)$.

$$F_{frict}\{v_{Slip}[\omega(t),v(t)],v(t)\} = \left\{ \begin{array}{c} \{v_{Slip}[\omega(t),v(t)] < v_{Fr}\} \cdot \mu_S \cdot \\ \arctan\left(\frac{v_{Slip}[\omega(t),v(t)]}{v_{Fr}}\right) \cdot \frac{4}{\pi} + \\ \{v_{Slip}[\omega(t),v(t)] \geq v_{Fr}\} \cdot \mu_K \cdot \frac{|v_{Fr}|}{v_{Fr}} \\ -\mu_{Lin} \cdot v(t) \end{array} \right\} \cdot F_N \qquad (4.39)$$

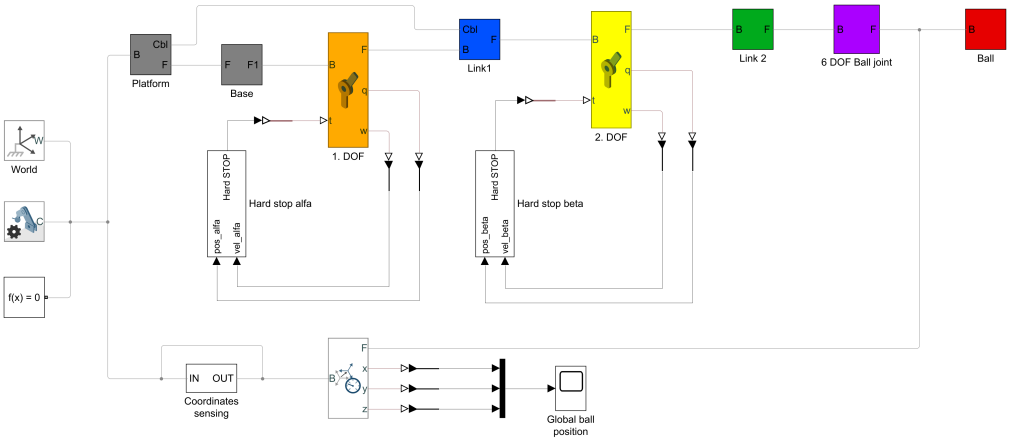
Napojení 6-DOF vazby na skript je v MATLABu řešen přes uživatelskou funkci a zapojení je na Obr. 4.59.

Další rozšíření fyzikálního modelu je v silových dorazech nakloněné roviny, které jsou v reálném systému řešeny jako gumové koncové dorazy. Toto omezení zamezí prolínání objemů jednotlivých částí modelu, což v reálném systému znamená, že mezi díly nedojde ke kolizi. Posledním rozšířením bylo přidání snímačů jak vůči globálnímu souřadnému systému, tak v dílčích lokálních souřadných systémech. Kompletní model, který už velice věrně napodobuje reálně postavený systém, je zobrazený na Obr. 4.60.

Po otevření bloku pro 6-DOF vazbu se otevře vyskakovací okno, v kterém je



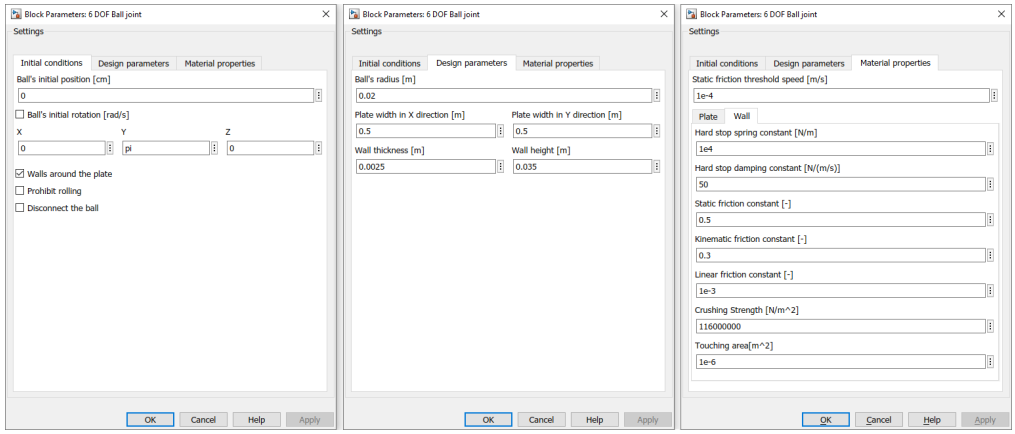
Obr. 4.59 Fyzikální model - rozšířená verze - část pro 6DOF vazbu kuličky



Obr. 4.60 Fyzikální model - rozšířená verze

implementována grafická maska pro parametrizaci všech údajů, které by mohlo být užitečné měnit - např. pro doladění podobnosti fyzikálního modelu s reálným systémem, změnu počátečních podmínek apod. Je tedy např. možnost nastavit počáteční podmínky pro simulaci, zrušit zábrany okolo roviny, zakázat kuličce rotaci a omezit ji tedy na 2-DOF (na úroveň zjednodušení matematického modelu), odstranit kuličku ze simulace, nastavit velikost roviny, kuličky i zábran, a

kompletně nadefinovat materiálové vlastnosti roviny i zábran a tím mít možnost ovlivnit jejich silové účinky na kuličku. Náhled na masku je na Obr. 4.61.

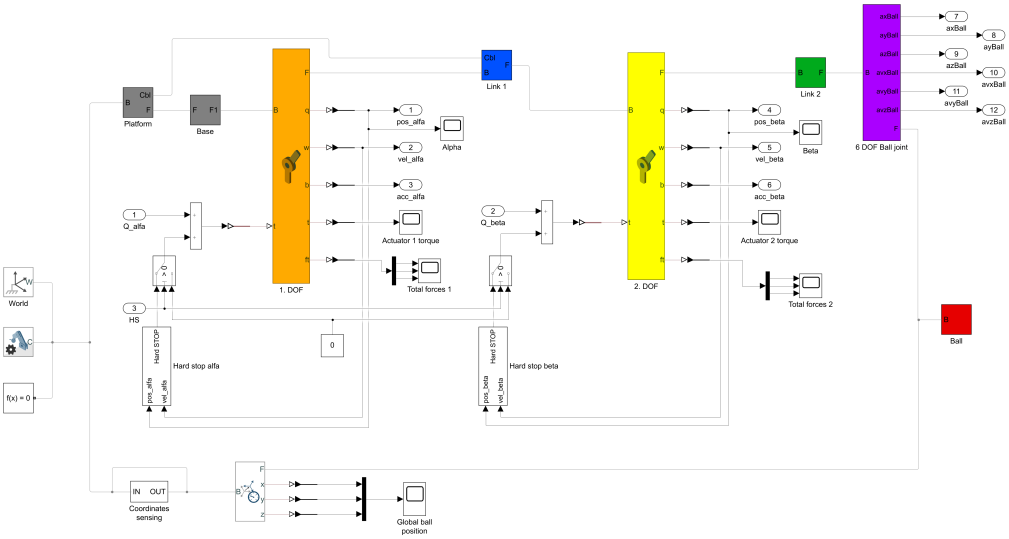


Obr. 4.61 Fyzikální model - parametrizace 6-DOF vazby

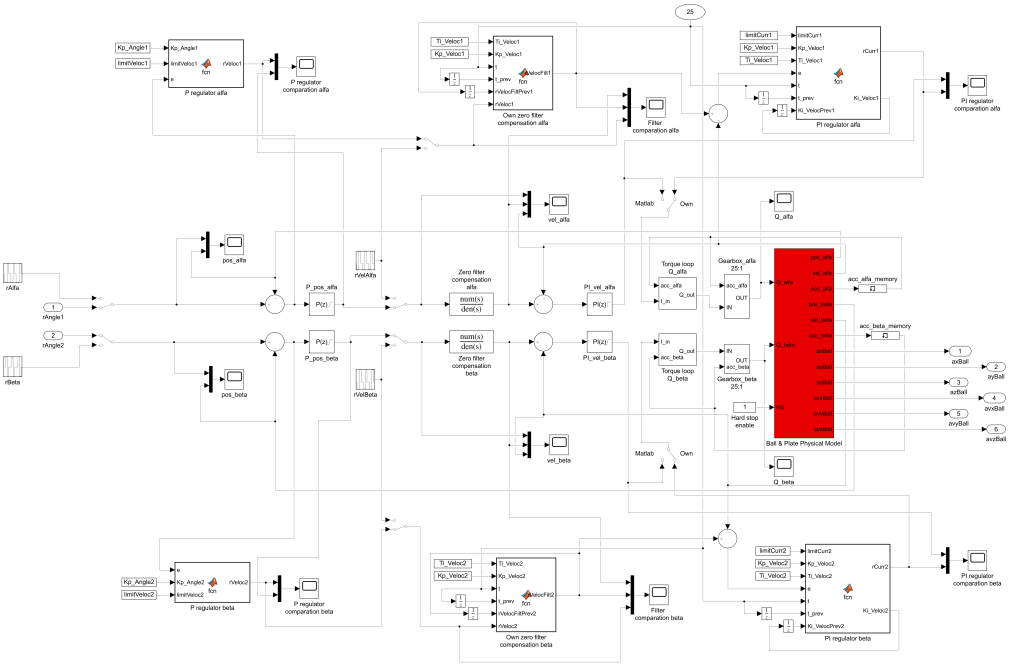
Celý fyzikální model je v angličtině, stejně jako např. názvy proměnných v řídicím programu apod. Je to z důvodu obecné srozumitelnosti a hlavně pro přípravu na publikace, kde byl a bude využíván (např. autorova publikace [P. 1] a [P. 5].

Rozšířená verze fyzikálního modelu je bez převodovek, náhrady ponechané proudové smyčky z řídicí jednotky motorů a bez řízení vyšších smyček. Finální verze fyzikálního modelu je uvedena na obrázcích 4.62, 4.63, 4.64 a zahrnuje mimo jiného tyto části:

- Kompletní regulace náklonu roviny včetně přepsané regulační struktury do kódu, s kontrolou totožného chování (nutné pro následné ladění a podobnost mezi fyzikálním modelem a reálným systémem).
- Kompletní regulace polohové smyčky pro kuličku, opět s přepisem do kódu pro implementaci do řídicího programu v C++.
- Fyzikální model použitých převodovek (setrvačné hmoty, účinnosti, tření apod.)
- Náhrada proudové smyčky identifikované z měření reálných proudových přechodových charakteristik na použitých motorech.
- Testovací průběhy pro typovou trajektorii kuličky - korespondují s reálným systémem.



Obr. 4.62 Fyzikální model - kompletní - základ pro regulaci



Obr. 4.63 Fyzikální model - kompletní - kaskáda pro regulaci náklonu roviny

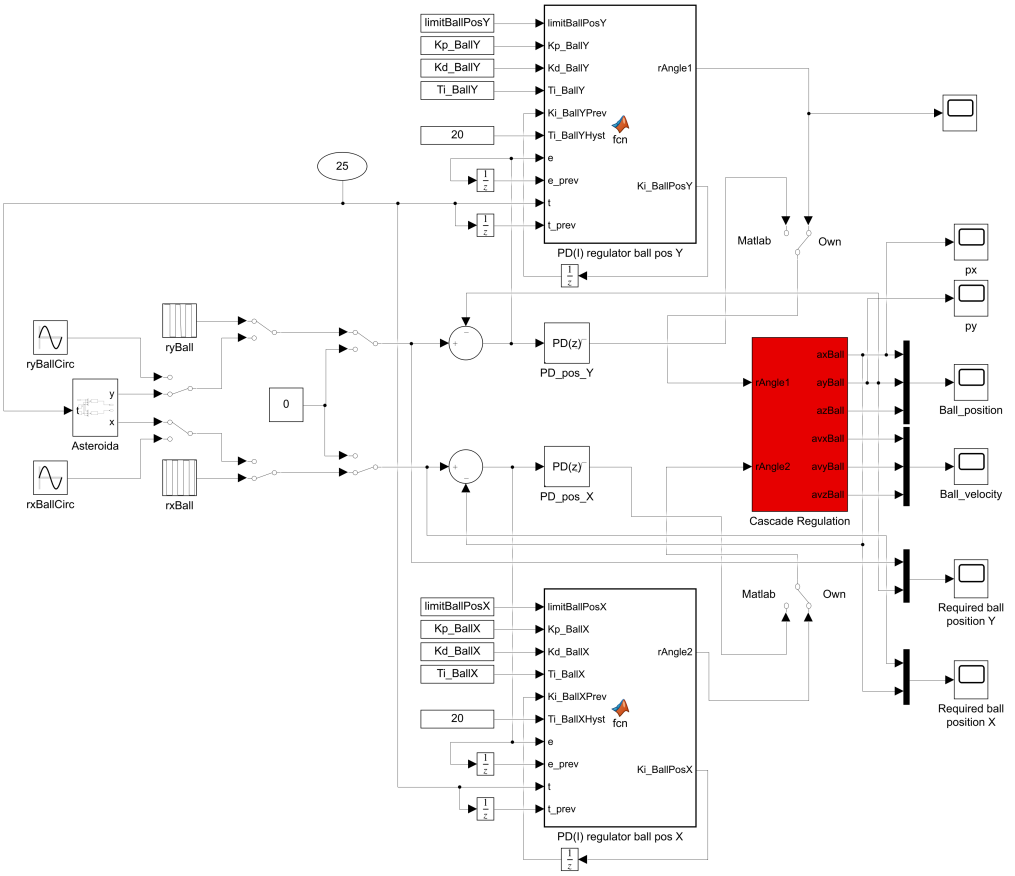
Na Obr. 4.62 je zobrazena nejnižší vrstva kompletního modelu a jedná se právě o část, která obsahuje fyzikální model. V porovnání s Obr. 4.55 došlo k minimálním změnám, byly pouze přidány vstupní a výstupní uzly + okna na grafické vykreslení průběhů pro následnou kontrolu, porovnání a vyhodnocení.

Na Obr. 4.63 je část modelu pro kaskádní regulaci náklonu roviny. Z důvodu přehlednosti byl z celého fyzikálního modelu z Obr. 4.62 vytvořen subsystém s názvem *Ball & Plate Physical Model*, který je vykreslen v červené barvě. Jeho středová část je složena ze standardních bloků pro PID regulátory, přenosové funkce apod. Pokud má ale fyzikální model sloužit k návrhu, testování a ladění regulátorů, které budou poté použity v reálném systému, je potřeba tyto regulátory umět přepsat do programovacího jazyka, v kterém je napsaný řídicí program. V opačném případě je celá regulační struktura dobrá pouze k simulačním účelům. Pro přepis a kontrolu správnosti přepsaných regulátorů slouží horní a spodní část modelu s uživatelsky definovanými funkcemi, v kterých jsou jednotlivé regulátory přepsány do podoby vhodné pro přímou interpretaci v C++. Příklad funkčního zápisu PI regulátoru rychlosti se saturací výstupního proudu je uveden ve Zdr. 28.

```
function [rCurr1, Ki_Veloc1] = fcn(limitCurr1, Kp_Veloc1,
    Ti_Veloc1, e, t, t_prev, Ki_VelocPrev1)
Ki_Veloc1 = Ki_VelocPrev1 + Kp_Veloc1 / Ti_Veloc1 * e * (t -
    t_prev);
rCurr1 = Kp_Veloc1 * e + Ki_Veloc1;
if rCurr1 > limitCurr1
    rCurr1 = limitCurr1;
elseif rCurr1 < -limitCurr1
    rCurr1 = -limitCurr1;
end
```

Zdr. 28 Zápis PI regulátoru rychlosti se saturací výstupního proudu

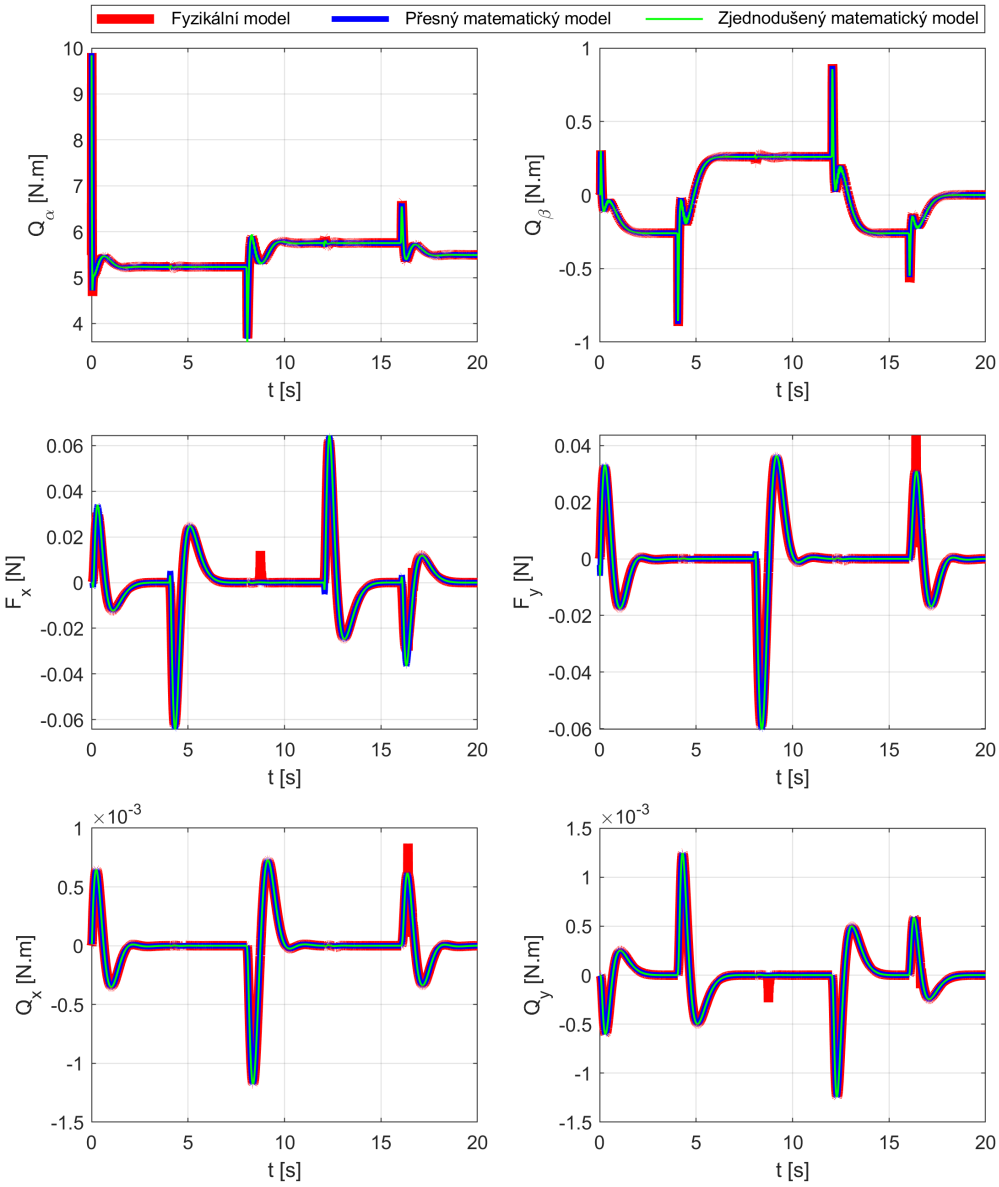
Nejvyšší vrstva regulačního obvodu je na Obr. 4.64 - jedná se o část pro polohovou regulační smyčku kuličky.
Pro přehlednost je opět celá kaskádní regulační struktura uzavřena do subsystému s názvem *Cascade Regulation*, který je rovněž vykreslen s červeným pozadím. Stejně jako u kaskádní regulace, i zde je středová část tvořena knihovními



Obr. 4.64 Fyzikální model - kompletní - regulace polohy kuličky

moduly regulátorů a horní i spodní část zobrazuje přepis regulátorů do zdrojových kódů. Na vstupu jsou různé testovací průběhy s typovou trajektorií (v čase se měnící žádaná pozice ve tvaru bodu, kružnice a hypocykloidy), na kterých budou probíhat testy a vyhodnocení.

Jak bylo na konci minulé kapitoly uvedeno, fyzikální model je pravděpodobně jedinou reálnou možností, jak otestovat správnost odvození matematického modelu. Je však potřeba zohlednit, že fyzikální model je mnohem složitější, má plných 8-DOF oproti 4-DOF matematického modelu, obsahuje tření, pružnost i tlumení mezi některými díly a v kloubech apod. Výsledné dynamické charakteristiky matematického a fyzikálního modelu tedy nebudou nikdy totožné, ani při zcela správně odvozeném matematickém modelu. Charakteristiky se vždy budou



Obr. 4.65 Porovnání průběhů matematického a fyzikálního modelu pro typovou trajektorii kuličky ve tvaru čtverce

mírně lišit, avšak trend dynamických průběhů musí být stejný. Výraznější změny jsou očekávány v situacích s vysokou dynamikou pohybu, kdy se bude zvyšovat

vliv z matematického modelu vyřazených, nebo ani nikdy nezohledněných částí původního modelu - zejména prokluz kuličky a z toho vyplývající silové/momentové špičky. Grafické porovnání zjednodušeného matematického modelu s modelem fyzikálním je zobrazeno na grafu v Obr. 4.65 a PŘÍLOHÁCH G a H.

Aby bylo možné oba modely porovnat, byl matematický model odvozen ještě pro 6-DOF namísto 4-DOF (rotace kuličky oddělena od translace a přepsána do dalších 2 pohybových rovnic). Jak bylo již dříve zmíněno, je to z důvodu nemožnosti přímo snímat kombinované silové účinky kuličky ve fyzikálním modelu. Způsob úpravy pro odvození tohoto 6-DOF matematického modelu je uveden v PŘÍLOZE B.

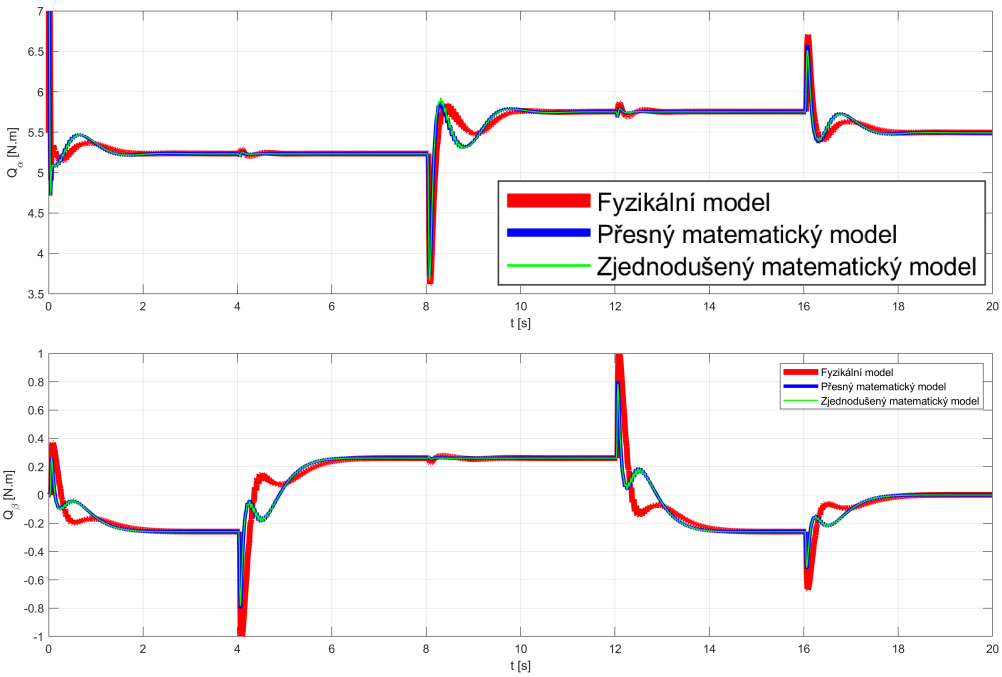
Stejně jako u porovnávání matematických modelů náleží testované průběhy typovým trajektoriím kuličky ve tvaru čtverce (Obr. 4.65), kružnice (PŘÍLOHA G) a asteroidy (PŘÍLOHA H). Průběhy odpovídají točivým momentům obou motorů (Q_α a Q_β), silovým účinkům kuličky v ose x a y (F_x a F_y) a točivým momentům kuličky okolo osy x a y (Q_x a Q_y). Jsou zde zobrazeny 3 trajektorie:

- Průběh s **červenou** barvou je získán z měření momentových a silových účinků na fyzikálním modelu, při požadavku na dodržení dané typové trajektorie kuličky.
- Průběh s **modrou** barvou náleží momentovým a silovým účinkům po dosazení průběhů stanových veličin a jejich derivací do přesného nezjednodušeného matematického modelu.
- Průběh se **zelenou** barvou náleží momentovým a silovým účinkům po dosazení průběhů stanových veličin a jejich derivací do přesného nezjednodušeného matematického modelu.

Z grafu je patrné, že byl matematický model odvozený správně, protože dynamické průběhy se až na pár silových/momentových impulzů překrývají. Tyto špičky, které jsou viditelné zejména v průbězích pro pohyb kuličky (především na grafech v PŘÍLOHÁCH G a H), jsou způsobené již zmíněnou redukcí počtu DOF u matematického modelu a absencí kombinovaného tření o podložku. Jakmile je kulička prudce brzděna změnou náklonu roviny, dojde k jejímu prokluzu na podložce a tento silový účinek odpovídá impulzům v dynamických charakteristikách. Podobně by tomu bylo i při odskocích kuličky od podložky, nebo při jejich nára-

zech o zábrany. Tyto vlivy nejsou v matematickém modelu zohledněny, protože by byl extrémně složitý, a pro účely ke kterým je odvozován, by byl nepoužitelný.

Podobnost fyzikálního a matematického modelu se začne výrazně lišit, zavedeme-li např. tření v kloubech, které je v reálných systémech nedílnou součástí každého pohyblivého spoje. Přesné určení takovéhoto tření je komplikované a muselo by se vždy porovnávat s reálným modelem, čímž by se fyzikální model více blížil k reálnému systému. Vliv zavedeného lineárního tření, jehož parametry byly experimentálně ověřeny na reálném systému, je demonstrován na průbězích v Obr. 4.65.

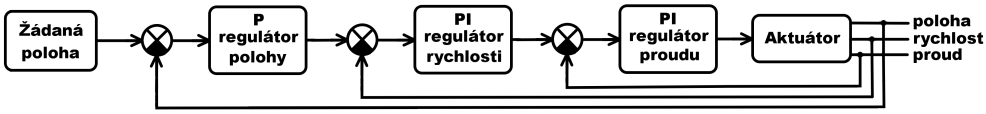


Obr. 4.66 Porovnání průběhů matematického a fyzikálního modelu pro typovou trajektorii kuličky ve tvaru čtverce při zavedení lineárního tření v kloubech

Oproti průběhům fyzikálního a matematického modelu v Obr. 4.65 se charakteristiky v Obr. 4.65 už začínají výrazněji lišit. Jde zejména o utlumení dynamických přechodů, což se při zavedení lineárního tření očekávalo. Výraznější změny lze pozorovat zejména u průběhu pro Q_β.

4.8.3 Kaskádní P(I)(D) regulace nakloněné roviny

Kaskádní regulace je stále nejrozšířenějším a dá se říci nejjednodušším typem regulace používané v průmyslových aplikacích řízení pohybu. Používá regulátory typu PID, se dvěma vnořenými smyčkami pro řízení rychlosti, a třemi vnořenými smyčkami pro řízení polohy. Parametry regulátorů se nastavují postupně od proudové smyčky, přes rychlostní, po polohovou. Optimální návrh je často složitý, při odvozování se běžně volí požadovaná doba ustálení přechodového děje a poté se simulačně ověřuje kvalita a robustnost navrženého způsobu řízení.



Obr. 4.67 Zjednodušený blokový diagram kaskádního uspořádání regulátorů

Běžně používaná struktura regulátorů je PI regulátor proudu, PI regulátor otáček a P regulátor polohy - viz Obr. 4.67. Platí, že systém může mít pouze takovou dynamiku, jak „ostře“ je nastavený proudový regulátor. Pokud je časová konstanta nadřazených smyček výrazně větší než časová konstanta proudové smyčky, nahrazuje se v dalším návrhu přenos proudové smyčky jedničkou, protože je v takovém případě tento přechodový děj z pohledu nadřazených smyček zanedbatelný [80].

Proudová smyčka

Jak už bylo několikrát zmíněno, proudová smyčka bude zachována, bude tedy ponechaná struktura regulátoru, který je součástí servozesilovače motorů. Protože je použitý PMSM, je motor řízen vektorově a složka I_q se podílí na vzniku točivého momentu. Proudová smyčka není nijak ovlivněna zátěží, která je dále na rotor motoru připojená, a závisí na konstrukčních a elektrických parametrech motoru. Z tohoto důvodu se předpokládá, že je její struktura navržena nejlepším možným způsobem, protože kdo jiný než výrobce ví, jak se jeho konkrétní typ motoru elektricky chová a jaké jsou jeho vlastnosti. Jedinou smysluplnou změnou by bylo nastavit „ostrost“ ponechaného regulátoru, tedy dynamiku přechodového děje proudu. K tomu je nutné mít možnost měřit momentotvorný proud na mo-

toru s dostatečně velkou vzorkovací frekvencí. K měření byl využitý ovládací a servisní software *TGZ GUI* umožňující nastavení a monitorování parametrů servozesilovače dodávaný výrobcem. Ten dokáže v režimu *Osciloskop* zaznamenávat průběhy až s frekvencí 20 kHz (podle zvoleného počtu měřených průběhů). Nastavitelné parametry proudového regulátoru přímo z programu TGZ GUI jsou uvedeny na Obr. 4.68.

Parametry

Jméno	Hod.osy 1	Hod.osy 2	Jednotka	Popis
Current controller				
C-K	500	500	mV/A	Current controller Q gain
C-Ti	500	500	µs	Current controller integral time
C-KDr	70	70	mV/A	Current controller D relative gain to K
C-Tc	1 000	1 000	µs	Time constant of current command filter
C-Filt	10	10	%	Percentable value of current command ...
C-LimN	3 000	3 000	mA	Current negative limit
C-LimP	3 000	3 000	mA	Current positive limit
C-VoltLimMin	90	90	% of PWM	Min voltage limit
C-VoltLimMax	90	90	% of PWM	Max voltage limit
C-CogCompFac	0	0	%	Cogging compensation factor

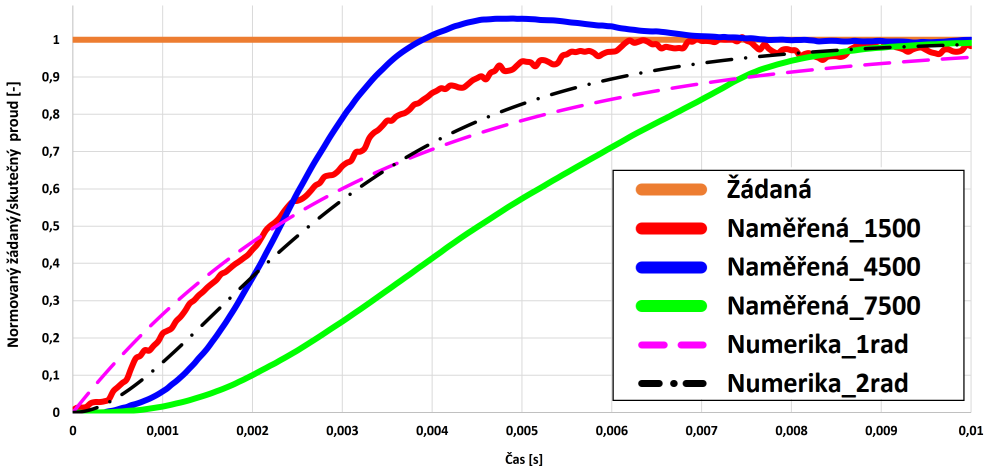
Obr. 4.68 Tabulka nastavení parametrů proudového regulátoru

Pro tyto účely budou pro nastavení dynamiky proudového regulátoru použity pouze parametry *C-K* a *C-Ti*, které jsou v obrázku zeleně orámovány. Konstanta *C-K* slouží k nastavení zesílení proudového regulátoru složky Q, konstanta *C-Ti* zase k nastavení integračního času proudového regulátoru. Další parametry nebudou měněny, protože jsou regulátorem buď ignorovány, nebo souvisí s vektorovým řízením, do kterého je riskantní zasahovat. Konkrétně to jsou:

- C-KDr - Zesílení proudového regulátoru složky D relativně k zesílení složky Q K (souvisí s vektorovým řízením)
- C-Tc - časová konstanta filtru žádaného proudu (nevyužitá)
- C-Filt - procentuální hodnota filtrované části žádaného proudu (100 = 100% filtrováno) (nevyužitá)
- C-LimN - omezení záporné amplitudy proudu relativně k povolenému trvalému efektivnímu proudu pro daný motor (na výstupu rychlostního regulátoru, který není v tomto případě použitý)

- C-LimP - stejné jako C-LimN, jen pro kladnou amplitudu proudu
- C-VoltLimMin - omezení záporného napětí na výstupu proudového regulátoru (souvisí s vektorovým řízením)
- C-VoltLimMax - stejné jako C-VoltLimMin, jen pro kladné napětí
- C-CogCompFac - faktor kompenzace coggingu: 0 = Vypnuto (před zapnutím musí být změřena kompenzační tabulka pro daný motor) - pozn.: cogging je parazitní složka momentu způsobená odlišností v geometrii statoru, způsobuje zvlnění momentu - [39]

Parametry *C-K* a *C-Ti* v Obr. 4.68 odpovídají továrnímu nastavení a je potřeba je přenastavit tak, aby byla dynamika přechodového děje proudu na maximální únosné hranici. Regulátor proudu v továrním nastavení pro motor v druhém DOF (nastavení úhlu β) odpovídá průběhům na grafu na Obr. 4.69.



Obr. 4.69 Výchozí nastavení proudového regulátoru pro motor v 2. DOF

Všechny charakteristiky byly vykresleny v časovém rozsahu 0 až 10 ms, aby byl evidentní vliv změny proporcionálního zesílení a integračního času proudového regulátoru. Měřený rozsah 0 až 7500 mA odpovídá nominálním proudům motorů, tedy jejich použitelnému proudovému rozsahu. Průběhy na Obr. 4.69 náleží normovaným přechodovým charakteristikám pro:

- skok proudu z 0 na 1500 mA - **červený** průběh

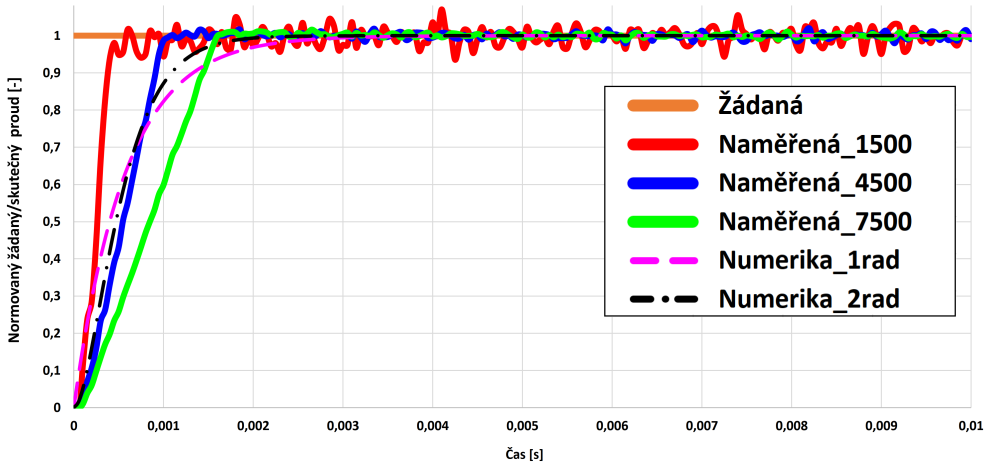
- skok proudu z 0 na 4500 mA - **modrý** průběh
- skok proudu z 0 na 7500 mA - **zelený** průběh
- aproximace gradientní numerickou metodou přenosem 1. řádu - **purpurový čárkovaný** průběh
- aproximace gradientní numerickou metodou přenosem 2. řádu se stejnými časovými konstantami - **černý čerchovaný** průběh
- normovaná žádaná hodnota - **oranžový** průběh

Přenosy aproximovaných přechodových charakteristik jsou uvedeny v rovnici 4.40. Časová konstanta odpovídá zhruba 3 ms a doba ustálení dosahuje cca 10 ms. Tyto časy jsou pro proudovou smyčku dosti velké a rychlost reakce proudu na změnu žádané hodnoty lze jistě ještě zvýšit, proto bude regulátor přenastaven pomocí experimentální Ziegler-Nicholsovy metody pro uzavřený regulační obvod.

$$G(s) = \frac{1}{0.00327 \cdot s + 1} \qquad G(s) = \frac{1}{(0.00157 \cdot s + 1)^2} \tag{4.40}$$

Při nastavení proudové smyčky musí být motor zabrzděn, aby nedocházelo k ovlivnění měřených přechodových charakteristik. Součástí struktury proudové smyčky, u které bylo rozhodnuto o jejím ponechání, jsou různá omezení, díky kterým nebudou měřené přechodové charakteristiky odpovídat učebnicovým příkladům. Mezi základní limity patří samozřejmě omezení proudu, který je schopna řídicí jednotka dodat. Kromě maximální povolené hodnoty proudu je také limitována jeho rychlost náběhu (tzv. „rampování“ proudu), vyplývající z konstrukce a typu motoru a zajišťující dodržení jeho předepsaných limitů. Další nedílnou součástí měření proudové odezvy motoru je vysokofrekvenční šum, který již byl z charakteristik vyfiltrován.

Při nastavování parametrů proudového regulátoru bylo dosaženo limitu (pomyslné meze stability), u kterého se dostával motor jen při minimálním vychýlení do rezonance. Přechodové charakteristiky jsou zobrazeny na Obr. 4.70 a odpovídající přenosy a nastavené konstanty regulátoru jsou v rovnici 4.41.



Obr. 4.70 Limitní nastavení proudového regulátoru pro motor v 2. DOF

$$C{-}K = 18\,000 \qquad C{-}Ti = 1\,150$$

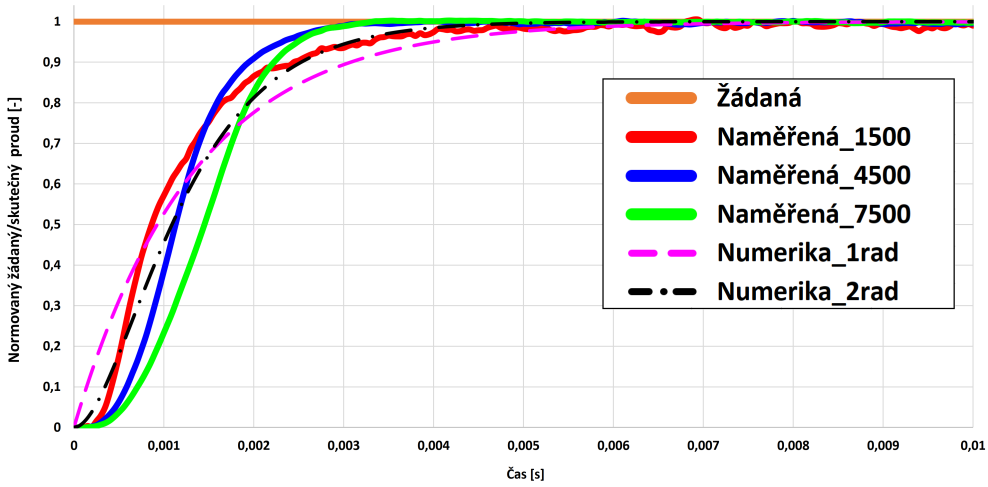
$$G(s) = \frac{1}{0.00058 \cdot s + 1} \qquad G(s) = \frac{1}{(0.00028 \cdot s + 1)^2} \tag{4.41}$$

Tento průběh je v reálné aplikaci nepoužitelný a musí být zatlumen. Čas ustálení je ale oproti výchozímu nastavení regulátoru cca 10x menší a hlavně je už viditelné dříve zmíněné „rampování“, což je hlavní důvod, proč je toto nastavení vůbec prezentováno. Při dalším ladění byla nalezena konfigurace, kterou lze považovat pro tyto účely za optimální. Je zobrazeno na Obr. 4.71 a parametry regulátoru spolu s přenosy jsou uvedeny v rovnici 4.42.

$$C{-}K = 2\,000 \qquad C{-}Ti = 950$$

$$G(s) = \frac{1}{0.00134 \cdot s + 1} \qquad G(s) = \frac{1}{(0.00065 \cdot s + 1)^2} \tag{4.42}$$

Přenosy uvedené v rovnici 4.42 neslouží pouze k dalšímu postupu při nastavování nadřazených smyček regulačního obvodu, ale také jako matematické vyjádření náhrady ponechané proudové smyčky ve fyzikálním modelu.



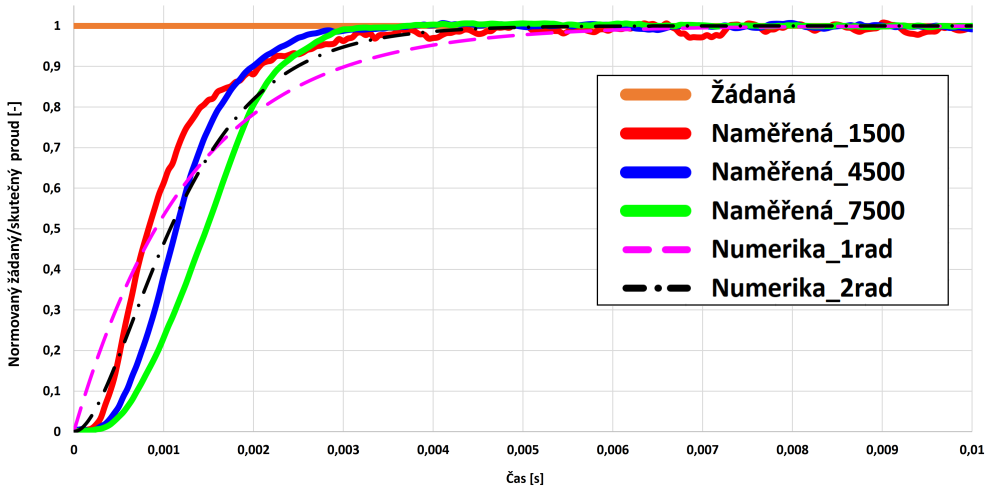
Obr. 4.71 Konečné nastavení proudového regulátoru pro motor v 2. DOF

Obdobný postup nastavení proudové smyčky byl aplikován i na motor v 1. DOF (nastavení úhlu α). Tento motor je totožný s motorem pro 2. DOF, pouze s rozdílem vyššího točivého momentu, což se týká až přepočtu za proudovou smyčkou (přesněji až na hřídeli motoru). Výsledné nastavení pro tento motor je na Obr. 4.72 a parametry regulátoru spolu s aproximovanými přenosy jsou v rovnici 4.43.

$$C\text{-}K = 4\,000 \qquad C\text{-}Ti = 1\,300$$
$$G(s) = \frac{1}{0.00131 \cdot s + 1} \qquad G(s) = \frac{1}{(0.00064 \cdot s + 1)^2} \tag{4.43}$$

Při porovnání přechodových charakteristik na Obr. 4.72 a 4.71 a přenosů v rovnicích 4.42 a 4.43 lze pozorovat pouze zanedbatelné rozdíly, nastavené parametry regulátorů se však výrazně liší. Podobnosti dynamických průběhů spolu s přenosy byly nastavena cíleně, protože je výhodné mít 2 různé motory, které mají však totožné chování i totožnou matematickou náhradu, s kterou se dále pracuje. Oproti tomu parametry proudových regulátorů jsou nastaveny pouze jednou a dále se již neřeší, proto jsou jejich konkrétní hodnoty nepodstatné.

Proudové regulátory jsou tedy nastaveny, bylo dosaženo jejich totožných dy-

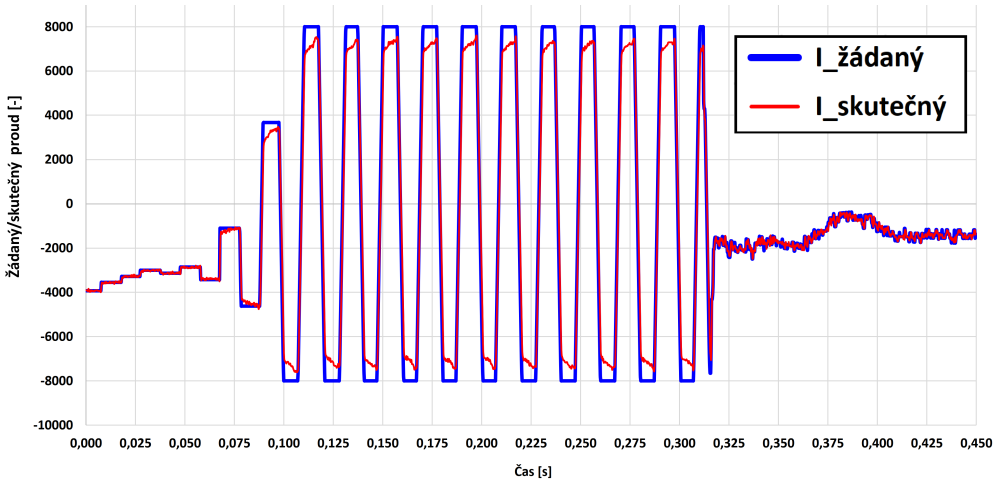


Obr. 4.72 Konečné nastavení proudového regulátoru pro motor v 1. DOF

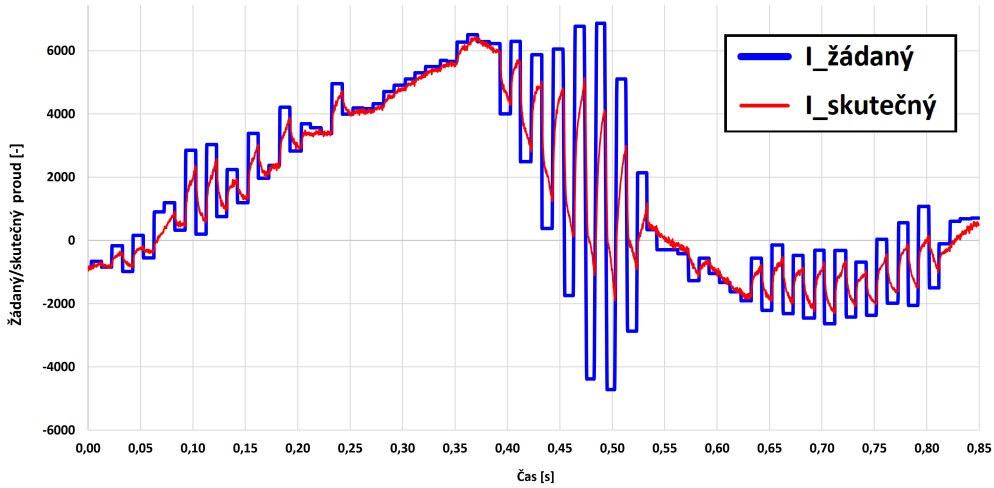
namických charakteristik i matematických popisů a hlavní zjištěný parametr je časová konstanta přechodového děje. Odpovídá zhruba **1,3 ms** a pokud bude z pohledu časových konstant nadřazených smyček zanedbatelná (v měřítku časové konstanty rychlostní smyčky se bude jevit jako skok), bude moci být celý přenos proudové smyčky nahrazen jedničkou. Tím by se mohl zjednodušit celkový popis systému - řád systému by poklesl o 1 nebo 2 stupně (podle plánované náhrady 1. nebo 2. řádem).

Stěžejním požadavkem na tuto vysokou dynamiku proudové smyčky je také vysoká vzorkovací frekvence žádané hodnoty proudu, která je na výstupu regulátoru rychlosti. V případě, že by byla rychlostní smyčka nastavena také ostře jako smyčka proudová a nebyl dodržený výše uvedený předpoklad, došlo by k rozkmitání žádaného proudu, který by proudový regulátor dokázal věrně kopírovat a tím by se systém dostal do proudové rezonance - Obr. 4.73 a 4.74.

Proudová rezonance odpovídá v podstatě rezonanci zrychlení, což vede v konečném důsledku k vysokofrekvenčním vibracím podložky (v porovnání s běžnými frekvencemi pohybu nakloněné roviny), a tedy, z pohledu dlouhodobého působení, k destruktivním mechanickým rázům.



Obr. 4.73 Proudová rezonance - příklad 1

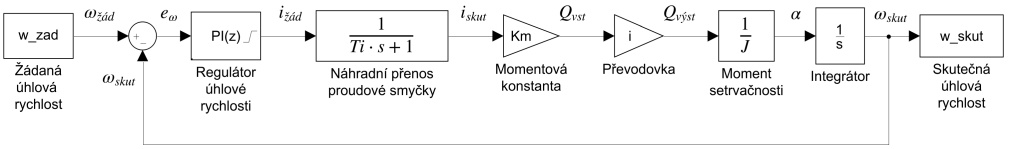


Obr. 4.74 Proudová rezonance - příklad 2

Rychlostní smyčka

Náhrada proudové smyčky ve tvaru aproximovaného přenosu 1. řádu byla implementovaná do fyzikálního modelu, a s omezeními vyplývajícími z parametrů jednotlivých motorů a servozesilovače byl model rozpohybován. Již z předchozích simulací, momentu setrvačnosti nakloněné roviny a z použitých převodovek se jeví, že budou časové konstanty rychlostní i polohové smyčky řádově vyšší,

než odezva proudové smyčky. To se však potvrdí až po odvození, nastavení a odsimulování rychlostního regulátoru.



Obr. 4.75 Blokové schéma kaskádní regulace pro nastavení rychlostní smyčky

Na Obr. 4.75 se nachází blokové schéma použité rychlostní smyčky, která je v kaskádní regulační struktuře nadřazená proudové smyčce. Regulační odchylka úhlové rychlosti je napojená na nastavovaný PI regulátor, jehož výstupem bude žádaný proud do již nastaveného PI regulátoru proudu, který je spolu s motorem nahrazen přenosem 1. řádu. Z něj vystupuje skutečný proud, který po vynásobení momentovou konstantou motoru a převodovým poměrem převodovky určuje výstupní točivý moment motoru. Ten je dále přiveden na zátěž, kterou reprezentuje moment setrvačnosti pro 1. DOF. Výstupem je úhlové zrychlení měřitelné na hřídeli převodovky, jehož integrací je dosažena skutečná úhlová rychlost. Přivedením změřené úhlové rychlosti přes zápornou zpětnou vazbou na vstup se určí regulační odchylka a celý postup se opakuje. Regulátor byl zvolený v paralelní formě, aby proporcionální složka neovlivňovala integrační složku a byl tak jasný vliv každé části regulátoru.

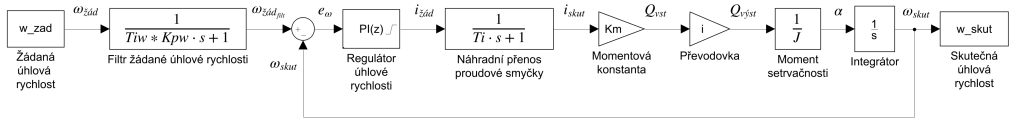
Jediným parametrem, který by se mohl v této chvíli jevit jako problematický na zjištění, je moment setrvačnosti pro nastavovaný DOF. Zde je s výhodou využitý **matematický model**, konkrétně matice D v rovnici 4.36. Její zjednodušená forma už obsahuje pouze momenty setrvačnosti jednotlivých DOF. Pro tuto strukturu regulátoru s pevně nastavenými parametry je vyžadováno, aby byl točivý moment přímo úměrný úhlovému zrychlení. Konstantou úměrnosti potom bude pouze moment setrvačnosti roviny, protože regulace roviny (ať už úhlové rychlosti, nebo náklonu) začíná z ustáleného stavu, kdy je rovina v mechanických dorazech (alespoň pro 1. DOF, u 2. DOF na výchozí pozici nezáleží, rovina je vůči ose rotace 2. DOF vyvážená), a neobsahuje kuličku. To znamená z matice D vyřadit ještě vliv kuličky - vynulováním její hmotnosti. Přenos otevřené smyčky má potom tvar:

$$\begin{aligned} G_{\omega_0}(s) = & \quad G_{PI}(s) \cdot \frac{1}{Ti \cdot s + 1} \cdot Km \cdot i \cdot \frac{1}{J} \cdot \frac{1}{s} \\ = & \quad \left(Kp_\omega + \frac{1}{Ti_\omega \cdot s} \right) \cdot \frac{Km \cdot i}{(Ti \cdot s + 1) \cdot J \cdot s} \end{aligned} \tag{4.44}$$

Přenos uzavřené smyčky s využitím Masonova pravidla má tvar:

$$G_\omega(s) = \frac{G_{\omega_0}(s)}{1 + G_{\omega_0}(s)} = \frac{\frac{Km \cdot i}{Ti_\omega \cdot Ti \cdot J} \cdot (Kp_\omega \cdot Ti_\omega \cdot s + 1)}{s^3 + \frac{1}{Ti} \cdot s^2 + \frac{Kp_\omega \cdot Km \cdot i}{Ti \cdot J} \cdot s + \frac{Km \cdot i}{Ti_\omega \cdot Ti \cdot J}} \tag{4.45}$$

V čitateli přenosu v rovnici 4.45 vznikla stabilní nula (minimálně fázový systém) ⇒ významný vliv derivace ⇒ odezva systému je složena z původní odezvy a násobku její derivace (mění odezvu mezi vstupem a výstupem) ⇒ větší překmit systému. Na regulační pochody má tedy nula negativní vliv, proto je odstraněna použitím filtru 1. řádu na žádané otáčky. Filtr bude v takovém tvaru, aby se nula vykrátila - časová konstanta filtru bude rovna časové konstantě otáčkové smyčky vynásobené proporcionálním zesílením. Přepracované blokové schéma je uvedené na Obr. 4.76 a výsledný přenos s odstraněním nuly v rovnici 4.46.



Obr. 4.76 Blokové schéma rychlostní smyčky kaskádní regulace s filtrem žádané rychlosti

$$G_\omega(s) = \frac{\frac{Km \cdot i}{Ti_\omega \cdot Ti \cdot J}}{s^3 + \frac{1}{Ti} \cdot s^2 + \frac{Kp_\omega \cdot Km \cdot i}{Ti \cdot J} \cdot s + \frac{Km \cdot i}{Ti_\omega \cdot Ti \cdot J}} \tag{4.46}$$

Pro nastavení rychlostního regulátoru bylo využito metody přiřazení pólů, při které se porovnávají koeficienty charakteristického polynomu přenosu uzavřeného regulačního obvodu s koeficienty charakteristického polynomu s 3-násobným kořenem - pro tento případ:

$$(\alpha + s)^3 = s^3 + 3 \cdot \alpha \cdot s^2 + 3 \cdot \alpha^2 \cdot s + \alpha^3 \tag{4.47}$$

Parametr α bude určen z Doddsova kritéria pro 5% dobu ustálení - [53, 20]:

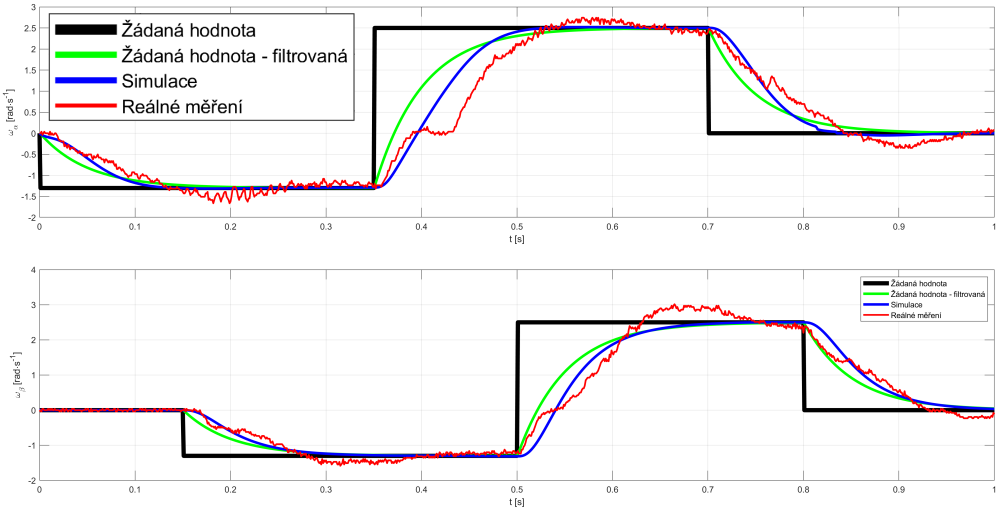
$$\alpha = 1.5 \cdot (n+1) \cdot \frac{1}{T_{ust}} \tag{4.48}$$

Kde T_{ust} je požadovaná doba ustálení přechodového děje a n označuje řád polynomu pro otáčkovou smyčku (tedy $n = 3$). Porovnáním koeficientů charakteristických polynomů u stejných mocnin jsou nalezeny vztahy pro nastavení rychlostního PI regulátoru.

$$Ti = \frac{1}{3 \cdot \alpha} \qquad Ti_\omega = \frac{Km \cdot i}{\alpha^3 \cdot Ti \cdot J} \qquad Kp_\omega = \frac{3 \cdot \alpha^2 \cdot Ti \cdot J}{Km \cdot i} \tag{4.49}$$

Kde Ti je časová konstanta nastavené proudové smyčky, Kp_ω náleží proporcionálnímu zesílení rychlostní smyčky a Ti_ω je časová konstanta integrace rychlostního PI regulátoru a také předřazeného filtru 1. řádu.
Na základě simulačního ověření a následných testů na reálném modelu byla zvolena doba ustálení $T_{ust} = 0.1\ s$. Porovnání výstupů ze simulace na fyzikálním modelu a reálných naměřených charakteristik je na Obr. 4.77
Rozsah otáček byl volen s ohledem na limity mechanických dorazů, aby bylo možné provézt testy a následné porovnání i na reálném systému. Úhlová rychlost je měřena za převodovkou a ustálené hodnoty odpovídají hraničním úhlům náklonu u mechanických dorazů. Dynamika rychlostní smyčky byla stanovena s ohledem na proudové limity motorů, dynamické požadavky, konstrukčně/mechanické možnosti modelu a dosažitelnou vzorkovací frekvenci žádané hodnoty proudu - viz proudová rezonance - Obr. 4.73 a 4.74. Dále byl standardní PI regulátor doplněn o výstupní saturaci proudu a omezení maximální hodnoty integrační složky kvůli naintegrování při proudové saturaci.

Naměřená data jsou zatížena především třením, vůlí v převodovce a šumem měření (úhlová rychlost ve zpětné vazbě je počítána numerickou derivací úhlu natočení), které v simulačním modelu obsažené nejsou. Vliv suchého tření je dobře viditelný při pohledu na naměřená data v okolí nulové rychlosti, zejména

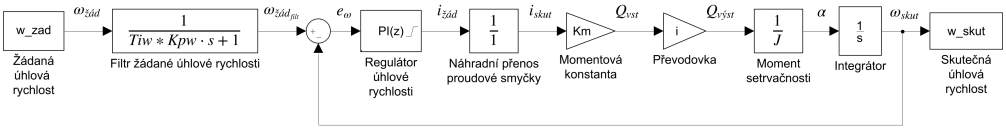


Obr. 4.77 Porovnání simulovaných a reálných přechodových charakteristik rychlostní smyčky - proudová smyčka nahrazena 1. řádem

u prvního DOF - setrvačnost zajistí rovině v okolí 0,4 s přejít na okamžik do kladných hodnot, ale při rychlosti blížící se nule musí motor překonat gravitaci i suché tření, což vede k pozastavení roviny (způsobené především účinky suchého tření).

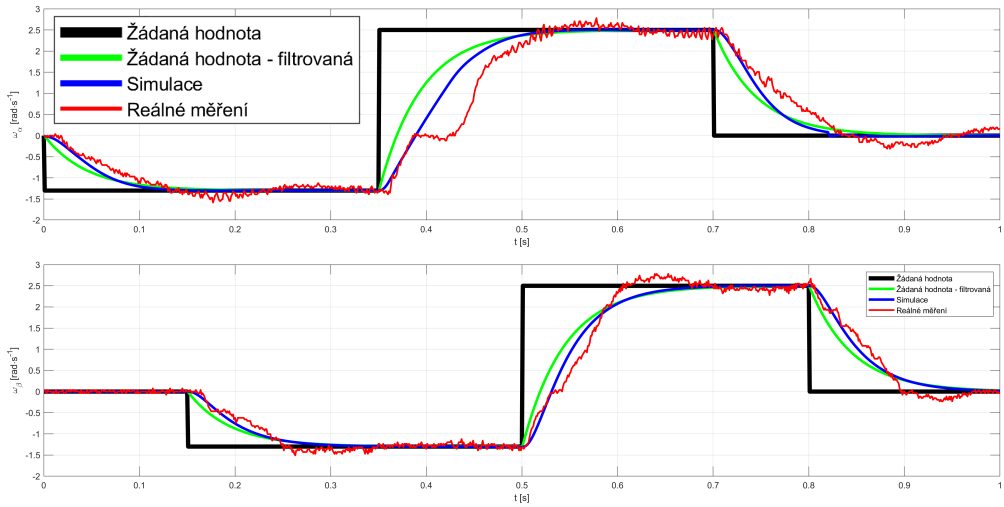
Z reálných přechodových charakteristik na Obr. 4.77 lze odhadnout časovou konstantu rychlostní smyčky, která se pohybuje v rozmezí 70 - 100 ms. I kdyby se však uvažovala časová konstanta filtrované žádané hodnoty rovná cca 50 ms, tak je oproti časové konstantě proudové smyčky stále řádově větší (cca 40x), oproti reálnému měření cca 60x. Z pohledu rychlostní smyčky je tedy dynamika (přechodový děj) proudové smyčky zanedbatelná a může být nahrazena jedničkou (předpoklad, že skutečná hodnota proudu dosahuje žádané hodnoty okamžitě). Tím se sníží řád přenosu regulačního obvodu a celá rychlostní smyčka bude přepsána do tvaru na Obr. 4.78 a v rovnicích 4.50 a 4.51.

$$G_\omega(s) = \frac{\frac{Km \cdot i}{Ti_\omega \cdot J}}{s^2 + \frac{Kp_\omega \cdot Km \cdot i}{J} \cdot s + \frac{Km \cdot i}{Ti_\omega \cdot J}} \tag{4.50}$$



Obr. 4.78 Blokové schéma regulace rychlostní smyčky po nahrazení proudové smyčky jedničkou

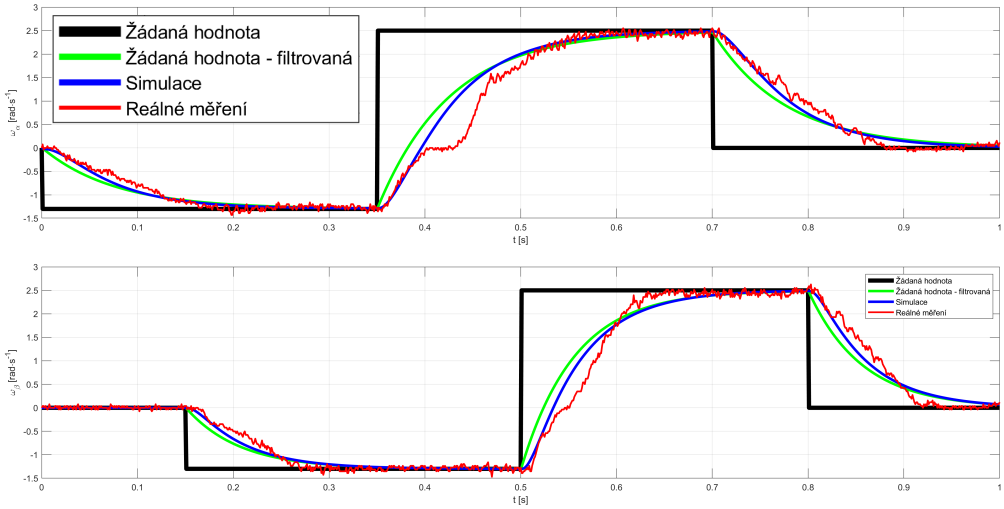
$$Ti_\omega = \frac{Km \cdot i}{\alpha^2 \cdot J} \qquad Kp_\omega = \frac{2 \cdot \alpha \cdot J}{Km \cdot i} \tag{4.51}$$



Obr. 4.79 Porovnání simulovaných a reálných přechodových charakteristik rychlostní smyčky - proudová smyčka nahrazena jedničkou

Zvolená doba ustálení je stejná jako u předchozí simulace, tedy $T_{ust} = 0.1\,s$. Jak je vidět na Obr. 4.79, regulační pochody jsou dosti podobné jako pro nastavení na Obr. 4.77, mají pouze mírně vyšší dynamiku, což je postřehnutelné zejména na průběhu úhlové rychlosti pro ω_β.

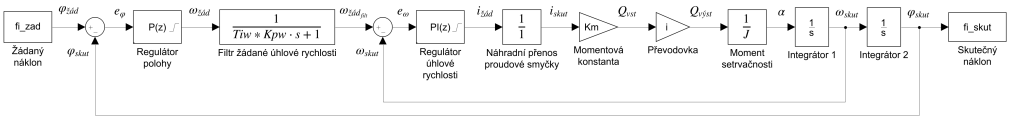
Po experimentálním doladění parametrů rychlostního PI regulátoru bylo dosaženo výsledků zobrazených na Obr. 4.80. Důraz byl kladen na co nejrychlejší dosažení žádané hodnoty (pokud možno bez překmitu), a s co nejmenším kmitáním okolo žádané hodnoty.



Obr. 4.80 Porovnání simulovaných a reálných přechodových charakteristik rychlostní smyčky - konečné experimentální doladění

Polohová smyčka

Na Obr. 4.81 je zobrazeno blokové schéma kaskádní regulace pro nastavení polohové smyčky, řídicí náklon roviny. Tato smyčka je nadřazena rychlostní smyčce a obsahuje pouze proporcionální regulátor, jehož vstupem je žádaná hodnota náklonu roviny. Polohová smyčky byla dále rozšířena o nastavitelnou periodu vzorkování P regulátoru, a to v násobcích vzorkovací periody rychlostní smyčky - tím může být uživatelsky doladěna rychlost její odezvy.



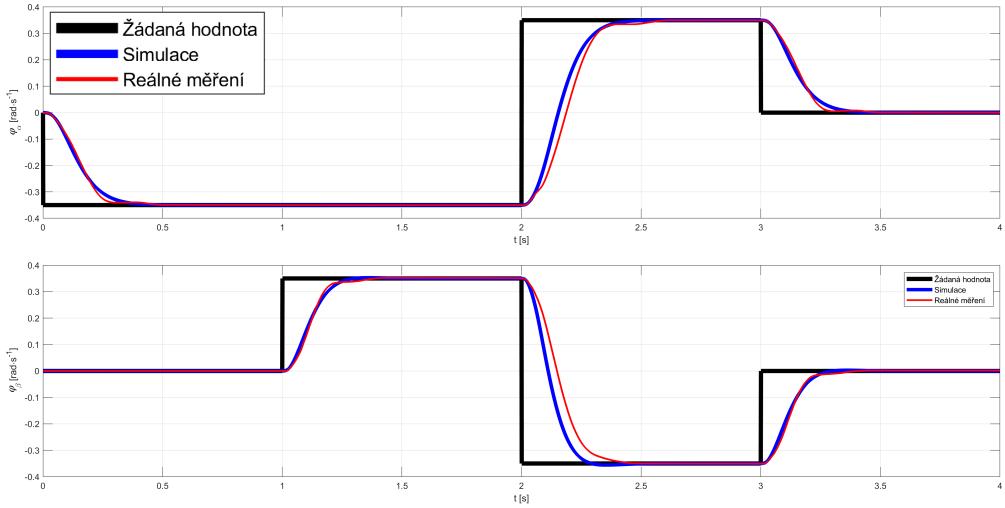
Obr. 4.81 Blokové schéma kaskádní regulace pro nastavení polohové smyčky

Přenos vyplývající z blokového schématu je uveden v rovnici 4.52 a parametry regulátorů v rovnici 4.53.

$$G_\varphi(s) = \frac{\frac{Kp_\varphi \cdot Km \cdot i}{Ti_\omega \cdot J}}{s^3 + \frac{Kp_\omega \cdot Km \cdot i}{J} \cdot s^2 + \frac{Km \cdot i}{Ti_\omega \cdot J} \cdot s + \frac{Kp_\varphi \cdot Km \cdot i}{Ti_\omega \cdot J}} \tag{4.52}$$

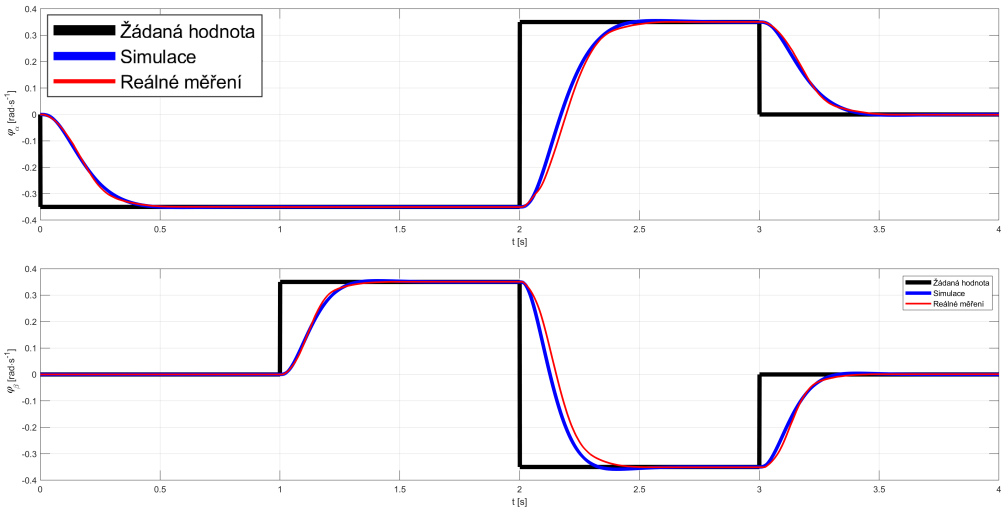
$$Kp_{\varphi} = \frac{\alpha^3 \cdot Ti_{\omega} \cdot J}{Km \cdot i} \qquad Ti_{\omega} = \frac{Km \cdot i}{3 \cdot \alpha^2 \cdot J} \qquad Kp_{\omega} = \frac{3 \cdot \alpha \cdot J}{Km \cdot i} \tag{4.53}$$

Doba ustálení polohové smyčky byla zvolena na $T_{ust} = 0.3\ s$. I mírně dynamičtější průběh se již rozkmitával, místy se dokonce dostával do rezonance a žádané maximální proudy byly již v limitních hodnotách. Na Obr. 4.82 jsou zobrazeny přechodové charakteristiky pro regulátory nastavené podle výpočtů v rovnici 4.53. Jsou zde opět dobře viditelné části ovlivněné suchým třením - téměř schodovitý průběh v částech charakteristiky s rychlostí blížící se 0.



Obr. 4.82 Porovnání simulovaných a reálných přechodových charakteristik polohové smyčky - základní nastavení

Po experimentálním doladění parametrů rychlostního PI regulátoru a polohového P regulátoru bylo dosaženo přechodového děje zobrazeného na Obr. 4.83. Testovací průběh byl stejný jako na Obr. 4.82, důraz byl opět kladen na co nejrychlejší dosažení žádané hodnoty bez překmitu. Dynamika přechodového děje byla mírně snížena, což ale přineslo plynulejší polohovou odezvu systému bez znatelných zákmitů.



Obr. 4.83 Porovnání simulovaných a reálných přechodových charakteristik polohové smyčky - konečné experimentální doladění

4.8.4 Regulace polohy/rychlosti kuličky

Řízení náklonu roviny je vyřešeno v předchozí podkapitole, nyní je konečně možné přistoupit k návrhu řízení polohy, popř. rychlosti kuličky. Celá kapitola 4.6 o zpracování obrazu, velká část kapitoly 4.7 o kinematických transformacích a v podstatě celá grafická aplikace a struktura navrženého systému včetně zapojení, směřuje k reálnému použití moderních metod strojového vidění. Objektem zájmu je právě řízení pozice/rychlosti kuličky na nakloněné rovině, přičemž nejde ani tak o princip tohoto řízení, jak o projevy komplexního chování celého systému. Reálná struktura má omezené možnosti, mezi které patří zejména frekvence, s jakou dokáže řídicí počítač zpracovávat snímky z kamery, nastavení a volba algoritmů zpracování obrazu, proměnné světelné podmínky, perioda vzorkování dílčích regulačních smyček, mechanické a silové možnosti celého systému a obecně odlišnost mezi matematickým/fyzikálním popisem a realitou. Mnohé ze zmíněného nelze dost dobře včlenit do simulačního modelu (byť tak detailního, jaký byt vytvořen), a velmi často je tato skutečnost opomíjena a nedoceňována. Jako výchozí testovací regulační struktura byla pro řízení polohy kuličky na nakloněné rovině vybrána PD(I) regulace.

PD(I) regulátor

PD(I) regulátorem je myšlen proporčně diferenční regulátor s automaticky připínatelnou integrační složkou. Ten bude na základě žádané pozice kuličky na nakloněné rovině, která do regulátoru vstupuje, generovat akční zásah v podobě žádaného náklonu roviny. Jak je z principu patrné, žádaná hodnota není řízena přímo, ale zprostředkovaně přes rozklad vektoru gravitační síly tak, aby jeho gradient zajistil očekávaný pohyb řízeného objektu - v tomto případu kuličky.

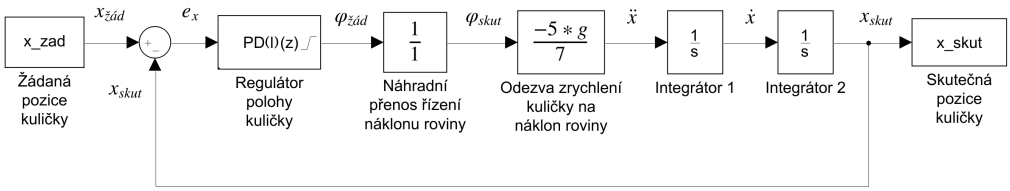
Podobně jako u kaskádní regulace náklonu roviny, i zde bude využit matematický model - 3. a 4. DOF v rovnicích 4.38 a 4.36. Pro zvolenou strukturu regulátoru s konstantními parametry je potřeba provést linearizaci v okolí zvoleného pracovního bodu a další zjednodušení matematického popisu. Linearizace bude probíhat v okolí horizontální polohy nakloněné roviny ($\alpha \approx 0; \beta \approx 0$), tedy $\sin(\alpha) \approx \alpha$. Jak při stabilizaci kuličky, tak při jejím řízení po žádané trajektorii, se rovina vždy pohybuje v okolí horizontálního náklonu. Co se týče zjednodušení, je nutné zanedbat vliv úhlového zrychlení roviny i vliv odstředivé síly, protože by parametry regulátoru nemohly být konstantní. Rovnice popisující zrychlení kuličky v obou osách mají po zjednodušení a linearizaci tvar:

$$\begin{aligned} \ddot{x} &= -\frac{5}{7} \cdot g \cdot \beta \\ \ddot{y} &= -\frac{5}{7} \cdot g \cdot \alpha \end{aligned} \tag{4.54}$$

Použitím Laplaceovy transformace na rovnici 4.54 je nalezena přenosová funkce popisující vztah mezi výstupní (x) a vstupní (α) veličinou:

$$\begin{aligned} G_\alpha(s) &= \frac{y(s)}{\alpha(s)} = -\frac{5}{7} \cdot \frac{g}{s^2} \\ G_\beta(s) &= \frac{x(s)}{\beta(s)} = -\frac{5}{7} \cdot \frac{g}{s^2} \end{aligned} \tag{4.55}$$

Z přenosů je parné, že má systém 2-násobný pól v nule, je tedy nestabilní a je nutné zavést regulaci, což je jasné i z pouhé fyzikální představy. Blokové schéma regulační struktury je uvedené na Obr. 4.84.



Obr. 4.84 Blokové schéma regulace kuličky pomocí PD(I) regulátoru

Podobně jako u rychlostní smyčky roviny v kaskádní regulaci (Obr. 4.75) je možné nahradit přechodový děj přenosu řízení celé kaskádní regulace přenosem 1. řádu, který vystihuje alespoň základní dynamiku odezvy skutečného náklonu roviny na žádaném úhlu natočení kloubů. Tento náhradní přenos by měl podle vyhodnocení přechodového děje na Obr. 4.83 časovou konstantu odpovídající cca $0.1\,s$. Celý přenos řízení s touto náhradou by byl 3. řádu a namísto požadované doby ustálení vyplývající z Doddsova kritéria, by v určení parametru α figurovala ona časová konstanta polohové smyčky náhradního přenosu. Bylo však výpočetně, simulačně i experimentálně ověřeno, že parametry regulátoru takto odvozeného modelu jsou použitelné pouze pro základní odhad a stejně musí být doladěny. Při porovnání je časová konstanta náhradního přenosu kaskádní regulace cca 8-10 x menší, než je časová konstanta přechodového děje pro regulaci kuličky. Na základě výše uvedeného je tedy zanedbána dynamika kaskádní regulace a nahrazuje se jednotkovým přenosem - viz. Obr. 4.84. Výsledkem bude přenos řízení 2. řádu s nastavitelnou dobou ustálení přechodového děje a koeficientem poměrného tlumení, kterým se bude omezovat vliv derivační složky regulátoru.

$$G_{xB}(s) = \frac{a \cdot (Kp_x + Kd_x \cdot s)}{s^2 + a \cdot Kd_x \cdot s + a \cdot Kp_x} \tag{4.56}$$

$$a = -\frac{5}{7} \cdot g \qquad Kd_x = \frac{2 \cdot \alpha \cdot \xi}{a} \qquad Kp_x = \frac{\alpha^2}{a} \tag{4.57}$$

Rovnice 4.56 a 4.57 jsou odvozeny pro polohovou smyčku kuličky v ose x, pro osu y je tvar výsledného přenosu totožný, liší se pouze nastavené parametry regulace. Doba ustálení přechodového děje byla zvolena pro obě osy odlišně, aby nebyla rychlejší smyčka (pro úhel β) omezena pomalejší dynamikou smyčky pro

úhel α. Konkrétně se jedná o hodnoty $Tx_{ust} = 2.2\ s$ a $Ty_{ust} = 3\ s$. Regulační pochody byly simulačně i experimentálně ověřeny pro 4 již dříve zmíněné typové trajektorie, kterými jsou:

- **Bod** - žádanou hodnotou je bod o souřadnicích $[x; y]$, který svou polohu v čase nemění. Ověřuje se schopnost regulátoru stabilizovat řízený objekt na přesně dané pozici, kdy všechny stavové proměnné kaskádní regulace náklonu roviny budou v ustáleném stavu nulové, stejně jako rychlost i zrychlení kuličky.

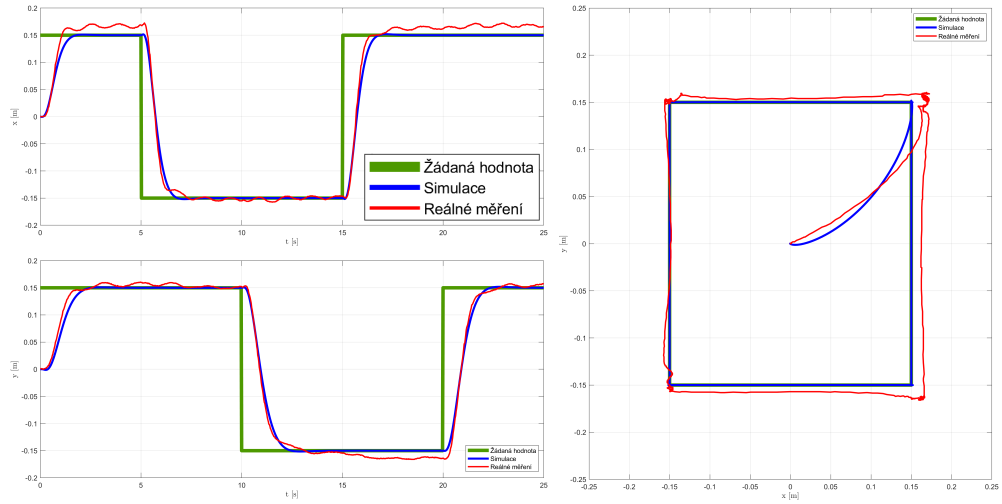
- **Čtverec** - podobně jako u předchozího případu je žádanou hodnotou bod, jehož pozice se však v čase skokově mění ve 4 různých konfiguracích odpovídajících po propojení čtverci. Ověřuje se, jak rychle dokáže regulátor stabilizovat řízený objekt v reakci na skok, a to po celé ploše nakloněné roviny. Obdobně jako pro bod jsou v ustáleném stavu všechny stavové proměnné kaskádní regulace náklonu roviny nulové, stejně jako rychlost i zrychlení kuličky.

- **Kruh** - žádanou hodnotou je trajektorie opisující kružnici, tedy bod, harmonicky měnící svou pozici v čase tak, aby udržoval od středu konstantní vzdálenost. Zde se zjišťuje, jak reaguje řídicí systém na harmonickou změnu žádané hodnoty pro různé frekvence. V ustáleném stavu (myšleno při ustálení rychlosti a dokonalém sledování žádané trajektorie), se všechny stavové proměnné kaskádní regulace náklonu roviny harmonicky mění, stejně jako stavové proměnné kuličky. Z pohledu LSS kuličky roste její poloha lineárně, její úhlová rychlost je tedy konstantní a její zrychlení nulové.

- **Asteroida** - reprezentuje maximální dynamiku změny všech stavových parametrů. Je speciálním případem hypocykloidy se 4 vrcholy, u které se žádaná pozice bodu na nakloněné rovině periodicky opakuje v křivkách, složených z více harmonických signálů. Tím je zajištěna dynamická změna všech stavových parametrů jak kaskádní regulace náklonu roviny, tak regulační smyčky kuličky. I z pohledu LSS kuličky se poloha, rychlost, i zrychlení dynamicky mění a žádná z nich není ani nulová, ani konstantní.

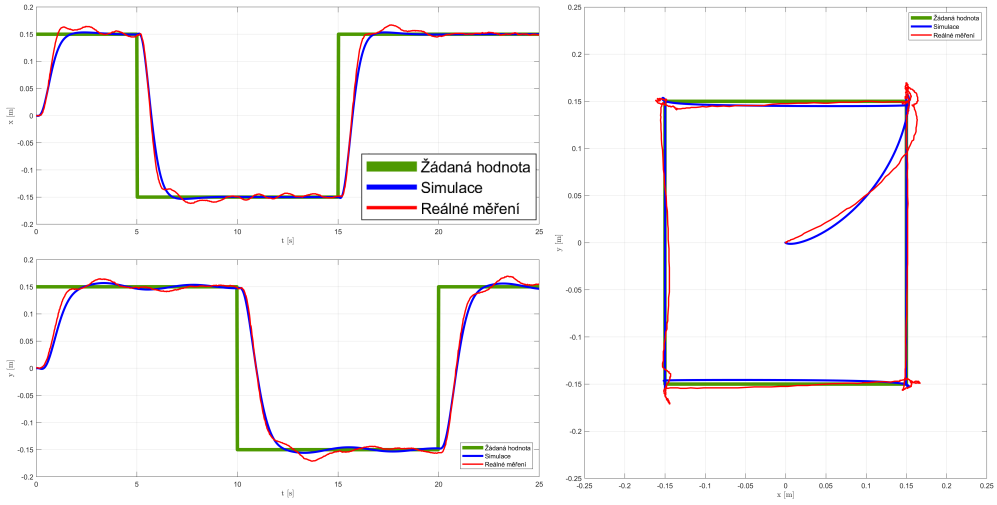
Grafické vyhodnocení reálně naměřených dat a ze simulace (fyzikálního modelu)

jsou uvedeny na grafech 4.85 - 4.88, ve kterých je vždy porovnáván čistý PD regulátor vyplývající z návrhu zákona řízení, s rozšířeným PD(I) regulátorem, u kterého je I složka připínána vždy až v blízkém okolí žádané hodnoty. V každém z grafů je reálná křivka určena mediánem z 10 nezávisle naměřených charakteristik a zobrazené průběhy náleží jak časové závislosti pozice kuličky v jednotlivých osách, tak $x - y$ závislosti reprezentující pohled na systém z perspektivy uživatele/kamery. Kvantifikované vyhodnocení kvality regulace bude uvedeno až při závěrečném porovnání různých zákonů řízení.

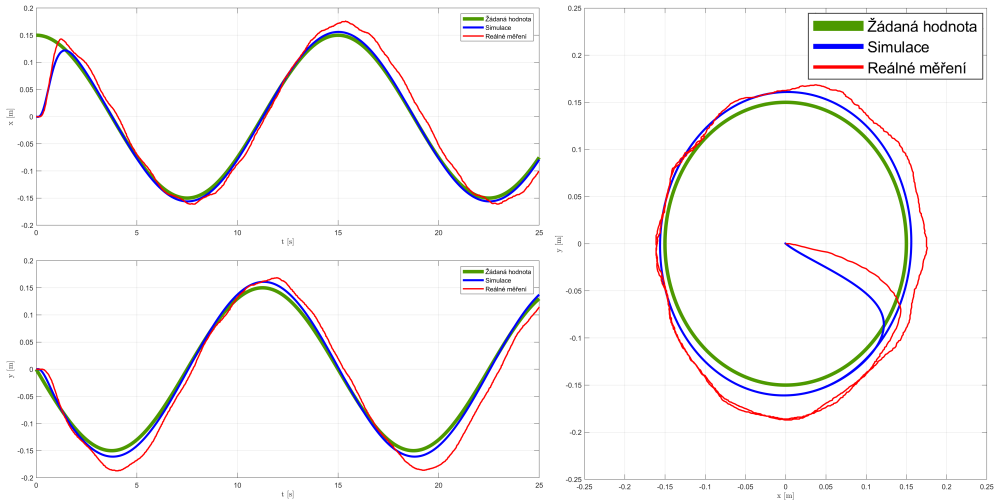


Obr. 4.85 Regulace kuličky - trajektorie typu Čtverec - PD regulátor

Na následujících grafech jsou zobrazeny dvě typové trajektorie, které ukazují chování řídicího systému v reakci na skokovou změnu žádané hodnoty (Obr. 4.85 a 4.86) a harmonickou změnu žádané hodnoty (Obr. 4.87 a 4.88). První z grafů vždy náleží PD regulátoru, druhý PD(I) regulátoru. Integrační složka v polohové smyčce je obecně problém, ale při jejím vhodném použití zajistí menší regulační odchylku, jak se vidět na obou grafech.

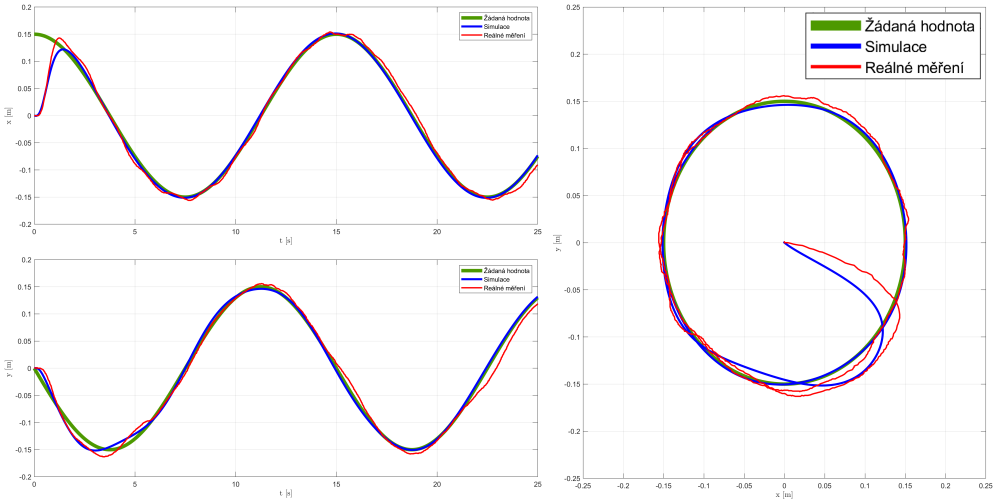


Obr. 4.86 Regulace kuličky - trajektorie typu Čtverec - PD(I) regulátor

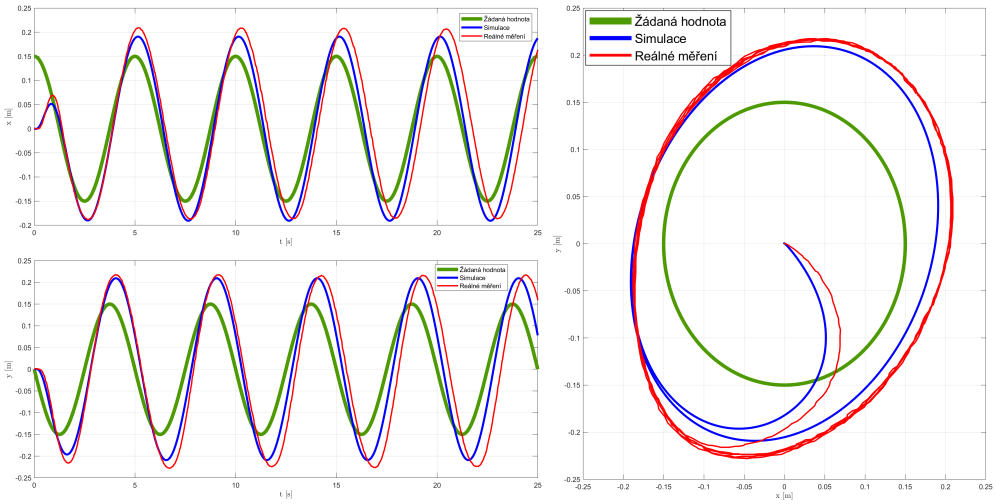


Obr. 4.87 Regulace kuličky - traj. typu Kruh - PD regulátor - perioda 15 s

PD(I) regulátor však není vhodný vždy, zejména při zvyšování rychlosti změny žádané hodnoty. Integrační složka potom zanese do řídicího systému pružnost, rychlost reakce na změnu žádané hodnoty se tak zpomaluje a celý systém se rozkmitá - viz Obr. 4.90. V takovém případě je lepší odepnout integrační složku a ponechat tedy čistý PD regulátor, i za cenu trvalé regulační odchylky - Obr. 4.89.

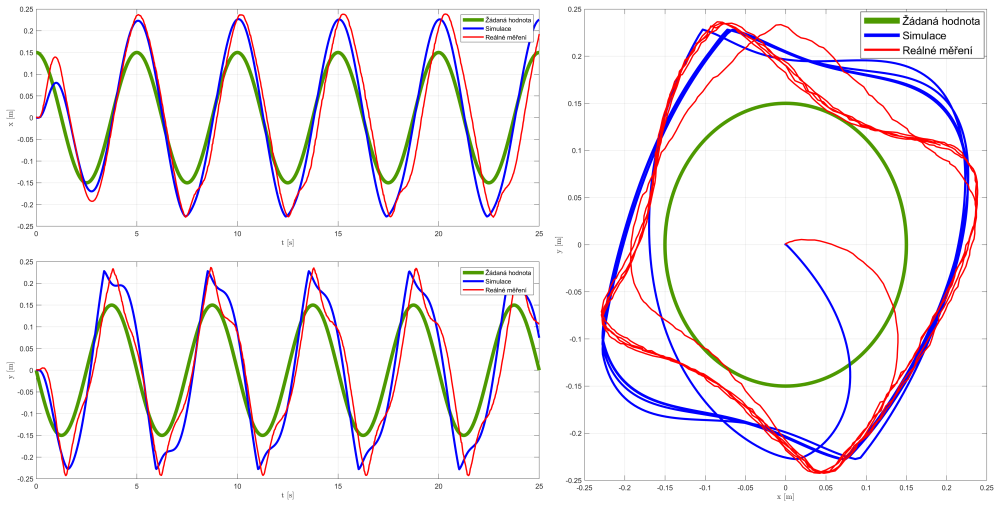


Obr. 4.88 Regulace kuličky - traj. typu Kruh - PD(I) regulátor - perioda 15 s



Obr. 4.89 Regulace kuličky - trajektorie typu Kruh - PD regulátor - perioda 5 s

Ostatní trajektorie typu Bod a Asteroida jsou uvedeny v PŘÍLOHÁCH I až L. Průběhy potvrzují, že pro pomalé změny žádané hodnoty je vhodné zavést integrační složku, pro rychlejší změny už je integrační složka více ke škodě, než k užitku.



Obr. 4.90 Regulace kuličky - trajektorie typu Kruh - PD(I) regulátor - perioda 5 s

4.8.5 Metoda výpočtu točivých momentů (computed torque)

Při dobré znalosti matematického popisu řízeného objektu je možné použít pro získání soustavy lineárních diferenciálních rovnic chyby řízení metodu výpočtu momentů (v anglické literatuře uvedenou jako *computed torque*), a na tu poté aplikovat techniky návrhu lineárních regulátorů. Tento typ regulace je založen na kombinovaném řízení v otevřené i uzavřené smyčce. Kvůli nepřesnostem v pohybových rovnicích, poruchám, změně parametrů modelu apod. existuje vždy rozdíl/odchylka mezi požadovanou a skutečnou trajektorií. K odstranění tohoto rozdílu je potřeba vždy zavést zpětnovazební řízení. Pro řídicí zobecněné síly je použit následující obecný zápis zákona řízení.

$$\underbrace{\mathbf{D}\left(\bar{q}\right)\cdot\ddot{\bar{q}}+\bar{H}\left(\bar{q},\dot{\bar{q}}\right)+\bar{G}\left(\bar{q}\right)}_{\bar{Q}_{ff}(t)}+\underbrace{\mathbf{D}\left(\bar{q}\right)\left[K_D\dot{\bar{e}}\left(t\right)+K_P\bar{e}\left(t\right)\right]}_{\bar{Q}_{fb}(t)}=\bar{Q}\left(t\right) \tag{4.58}$$

Matice K_D a K_P jsou diagonální matice konstantních zesílení typu $n \times n$, K_D má význam vlastních frekvencí, K_P má význam tlumení. Výsledný zákon řízení momentů má dvě složky.

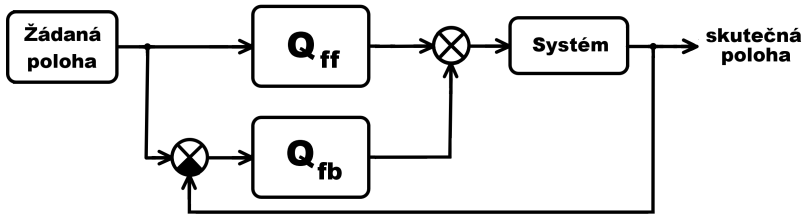
Složka $\bar{Q}_{ff}(t)$ představuje přímé řízení (feedforward) - výpočet zobecněných sil (akčního zásahu) ze známého matematického modelu a žádané hodnoty. Při dokonale přesném matematickém popisu a realizovatelnosti vypočítaných momentů, by nevznikla žádná odchylka a tato složka by tedy postačovala pro zajištění přesného sledování žádané trajektorie.

Určitá odchylka ale vznikne vždy, a momentové požadavky je mnohdy nereálné splnit, proto je zde složka $\bar{Q}_{fb}(t)$ představující řízení ve zpětné vazbě (feedback) - výpočet korekčních zobecněných sil minimalizující regulační odchylku, tedy chybu mezi žádanou a skutečnou trajektorií.

Po dosazení rovnice 4.58 do obecné pohybové rovnice a následné úpravě je výsledkem soustava lineárních diferenciálních rovnic pro rozdíl mezi žádanými a skutečnými hodnotami řízeného objektu, viz rovnice 4.59.

$$\ddot{\bar{e}}(t) + K_D\dot{\bar{e}}(t) + K_P\bar{e}(t) = \bar{0} \tag{4.59}$$

Aby bylo možné tuto úpravu realizovat, musí být matice D regulární (kvůli její inverzi) a žádaná trajektorie musí být alespoň dvakrát derivovatelná. Regularita matice D je fyzikální vlastností systému, ale 2x derivovatelný řízený vstup je nutné dostupnými prostředky zajistit. Výhoda tohoto řízení by mohla být při předem známé žádané hodnotě a možném offline výpočtu odpovídajících momentů. Nelinearita se poté v přímovazební části odečte a vzniklá chyba mezi matematickým popisem a realitou se doreguluje zpětnovazebním regulátorem.



Obr. 4.91 Zjednodušený blokový diagram metody výpočtu momentů

Matice jsou však závislé na stavových veličinách, které nejsou většinou předem

známé, proto je online výpočet přímovazebních i zpětnovazebních momentů nezbytný. Je tedy nutné ověřit, jestli bude řídicí počítač schopný vykonávat tento přepočet v reálném čase a zajistit tak dostatečnou frekvenci aktualizace žádaných hodnot pro proudovou smyčku [80, 60, 79].

Rozvaha nad konkrétním zadáním

Pro tento konkrétní případ bude regulátor typu MIMO, protože bude založen na odvozených pohybových rovnicích uvedených v postupu v PŘÍLOZE C a v rovnici 4.35, které jsou mezi sebou provázané stavovými proměnnými. Nejdříve bude uvažován originální přesný matematický model, poté jeho zjednodušená verze, a ověří se, jestli má ono zjednodušení významný vliv na kvalitu regulačních pochodů.

Žádaná trajektorie je generována buď z nadřazeného polohového regulátoru kuličky (kapitola 4.8.4), nebo zadána v libovolné formě uživatelem. V obou případech má charakter skokové změny a je nutné zajistit její transformaci do alespoň 2x derivovatelného tvaru, aby mohl být tento zákon řízení aplikovatelný. K tomu bude použita tzv. „průmyslová/filtrovaná derivace“ [80], protože v reálných podmínkách je generování signálu přímé derivace problematické a je nutné ji vhodným způsobem rekonstruovat. Vzorkovací frekvence polohové smyčky řízení kuličky je navíc omezena zejména vzorkovací frekvencí použité kamery, pohybuje se okolo 40 Hz/FPS, což je pro řízení pohybu dosti málo, a její velké skokové změny by mohly dostat systém do proudové rezonance. Při využití stávající regulační struktury kuličky bude tedy pravděpodobně nutné zavést filtraci žádané hodnoty, která zajistí její plynulejší změnu, byť za cenu mírně posunuté fáze.

Z rovnice 4.59 je patrné, že je také nutné získat derivace skutečných hodnot (pro výpočet regulační odchylky), které mají také charakter skokových změn s frekvencí mezi vzorky odpovídající vzorkovací frekvenci řídicího vlákna. Proto i zde bude použita dříve zmíněná průmyslová derivace. Dále je zřejmé, že zpětnovazební regulátor má v podstatě pouze proporcionální a derivační složku, a tedy nezajišťuje nulovou regulační odchylku polohy. Integrační složku polohy není ale možné zařadit v celém rozsahu regulace, protože by systém rozkmitala, a musí se tedy volit obdobný způsob, jako v kapitole 4.8.4 - automatické připínání s hysterezí.

Aby byla využita v maximální míře simulace na fyzikálním modelu pro naladění tohoto experimentálně zavedeného regulátoru, je vhodné simulovat všechny vlivy, které se v reálném systému vyskytují a mohly by mít negativní vliv na jeho výsledné chování. Z principu použité metody je zásadním problémem šum snímaného signálu (úhly náklonu roviny), který je sice na první pohled zanedbatelný (jednotky promile z celého rozsahu), ale po jeho opakovaném derivování by mohl mít zásadní dopad na celý regulační pochod. Bude tedy experimentálně změřen a zaveden do simulace ve formě bílého šumu.

Veškeré limity, které reálný systém obsahuje, musí být taktéž v simulačním modelu uvažovány, a jsou jimi především:

- limit točivého momentu - proudové omezení řídicí jednotky
- limit vzorkovací frekvence regulátoru - závisí na složitosti jeho algoritmu, výpočetním výkonu a vytíženosti řídicího počítače, rychlosti jeho komunikace mezi servozesilovačem, atd.
- limit vzorkovací frekvence nadřazených smyček - např. při řízení kuličky, kdy je žádaná hodnota generována s omezením FPS kamery
- limit dosažitelnosti žádaných zrychlení/rychlostí/poloh - vztah mezi reálně dosažitelným točivým momentem a hmotou/setrvačností celé sestavy

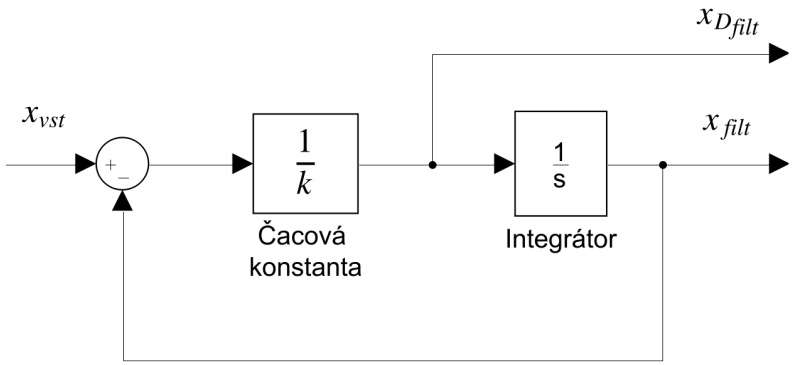
Vše výše zmíněné vychází z detailní analýzy celého reálného modelu, a při opomenutí byť jediného bodu nebude reálné chování systému korespondovat s chováním simulačního modelu. Tento typ regulátoru předpokládá dokonalou znalost řízené soustavy, jak z pohledu jejího chování a omezení, tak z pohledu matematického vyjádření dynamiky systému - alespoň pro jeho blízké okolí.

Průmyslová/filtrovaná derivace

Využívá integrálu v kombinaci s filtrem 1. řádu s nastavitelnou časovou konstantou. Integrál slouží jako paměť předchozí hodnoty, filtr zase tlumí šumy a omezuje velikost počátečního impulzu při skokové změně žádané hodnoty. Tato kombinace umožňuje opakovaně derivovat zašuměný signál, u kterého klasická diference selhává. Filtrace probíhá až na výstupu, takže neposouvá fázi, a tedy nijak výrazně nezkresluje odhad derivace. Vhodným nastavením časových konstant je potom možné přizpůsobit požadovanou dynamiku systému dynamice,

které je reálně dosažitelná, a zvýšit tak vliv přímovazební části regulátoru.

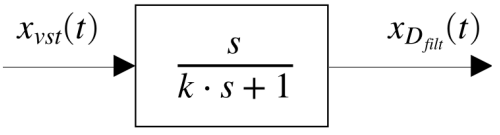
Na Obr. 4.92 je znázorněno blokové schéma průmyslové/filtrované derivace. Pro takto vytvořené zapojení platí rovnice 4.60 a Obr. 4.93.



Obr. 4.92 Blokové schéma rekonstrukce derivace signálu

$$X_{filt}(s) = \frac{X_{vst}}{k \cdot s + 1} \qquad X_{D_{filt}}(s) = \frac{X_{vst}(s) \cdot s}{k \cdot s + 1} \tag{4.60}$$

Pro přepis do diskrétní oblasti tak, aby šel algoritmus matematicky zapsat do programu, bude derivace nahrazena tangensem sečny grafu tvořené dvěma body vzdálenými od sebe o časový úsek h (tento úsek odpovídá periodě vzorkování regulátoru).



Obr. 4.93 Výsledný blok pro rekonstrukci derivace signálu

$$\frac{d\,x_{filt}(t)}{dt} = \frac{1}{k} \cdot [x_{vst}(t) - x_{filt}(t)]$$

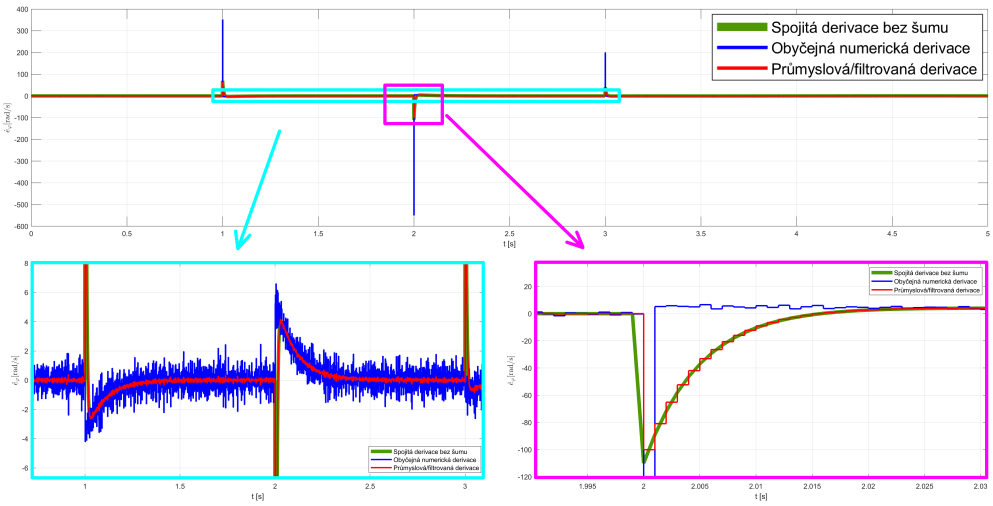
$$\frac{d\,x_{filt}(t)}{dt} \approx \frac{x_{filt}(t+h) - x_{filt}(t)}{h}$$

$$\frac{1}{k} \cdot [x_{vst}(t) - x_{filt}(t)] \approx \frac{1}{k} \cdot \frac{x_{vst}(t) - x_{filt}(t) + x_{vst}(t+h) - x_{filt}(t+h)}{2} \tag{4.61}$$

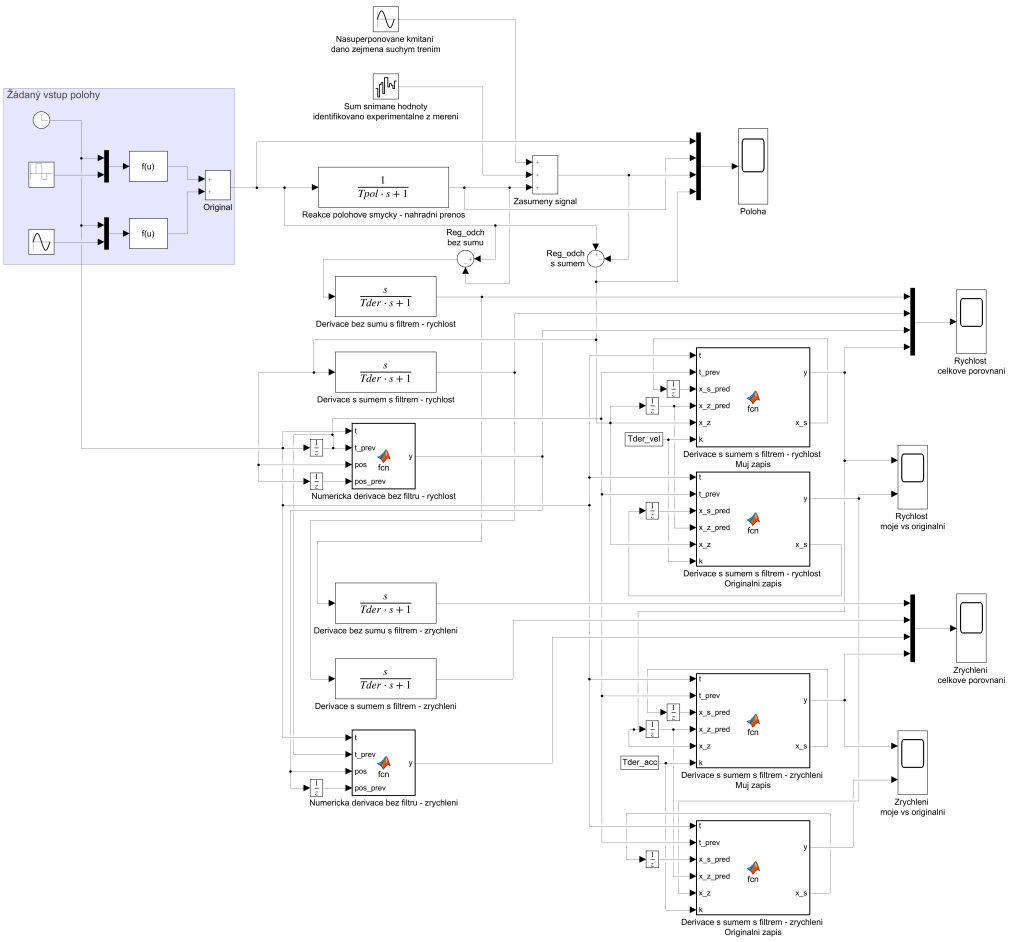
$$x_{filt}(t+h) = x_{filt}(t) \cdot \frac{2 \cdot k - h}{2 \cdot k + h} + \frac{h}{2 \cdot k + h} \cdot [x_{vst}(t) + x_{vst}(t+h)]$$

$$x_{D_{filt}}(t+h) = \frac{1}{k} \cdot [x_{vst}(t+h) - x_{filt}(t+h)]$$

Rovnice 4.61 byly nejdříve přepsány do MATLABu/SIMULINKu a jejich chování odsimulováno na typových průbězích, na které byl nasuperponovaný měřený šum. Vše bylo porovnáno jak s originálním zápisem průmyslové/filtrované derivace, tak s běžnou numerickou a spojitou derivací. Schéma zapojení je zobrazeno na Obr. 4.95.



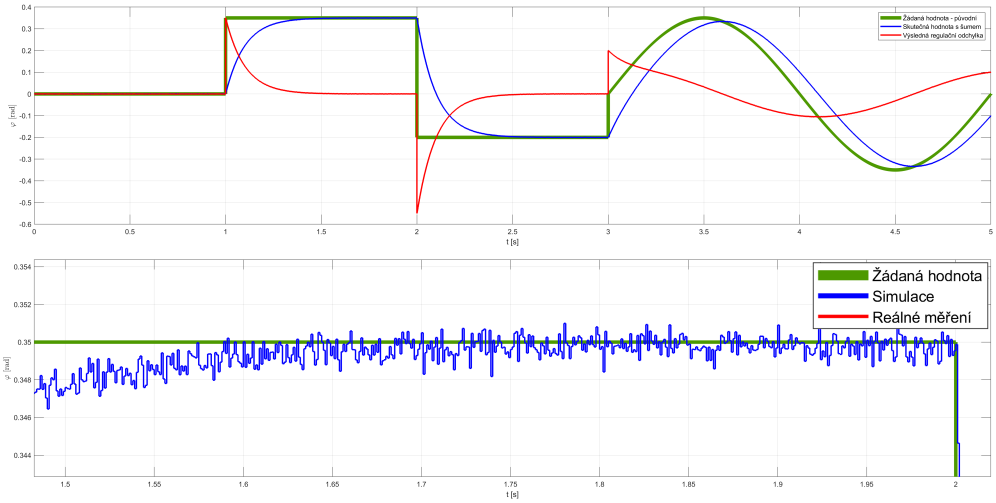
Obr. 4.94 Porovnání spojité, diskrétní a průmyslové/filtrované derivace - rychlost



Obr. 4.95 Simulační schéma pro testování rekonstrukce derivace

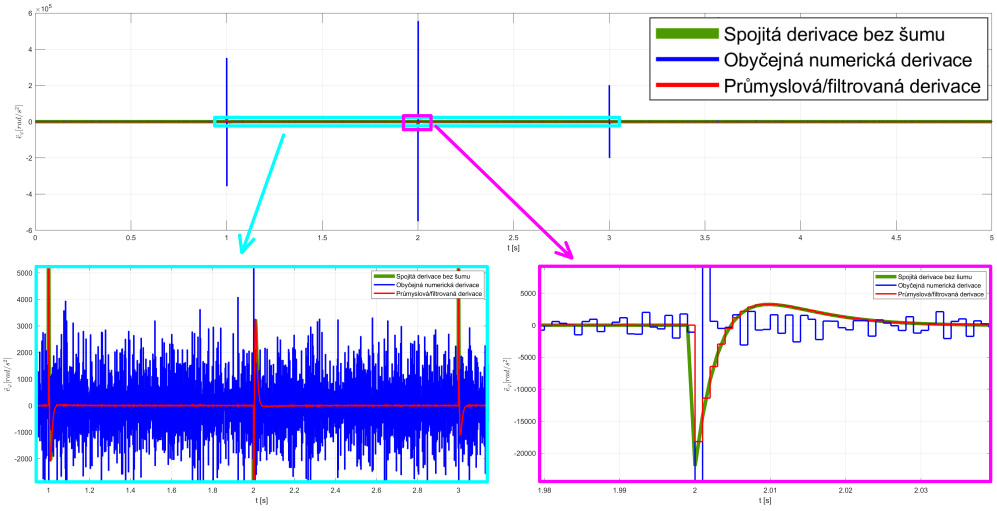
V Grafech na Obr. 4.94 a 4.97 jsou porovnány průběhy 1. a 2. derivace regulační odchylky polohy z Obr. 4.96. Snímaná poloha byla pro testy simulována jako odezva na skokovou a harmonickou změnu žádané hodnoty s přechodovým dějem 1. řádu, na kterou byl nasuperponován šum s parametry odpovídajícími měření na reálném systému - kombinace harmonického signálu o frekvenci 0.3 Hz a amplitudě 0.35 mrad, s bílým šumem o spektrální výkonové hustotě 25 pW/Hz a korelačním čase 100 µs.

Graf je zobrazen vždy pro celý rozsah, a poté je pro lepší přehlednost průběh přiblížen na významný detail. Na Obr. 4.96 lze až při velkém přiblížení vidět za-



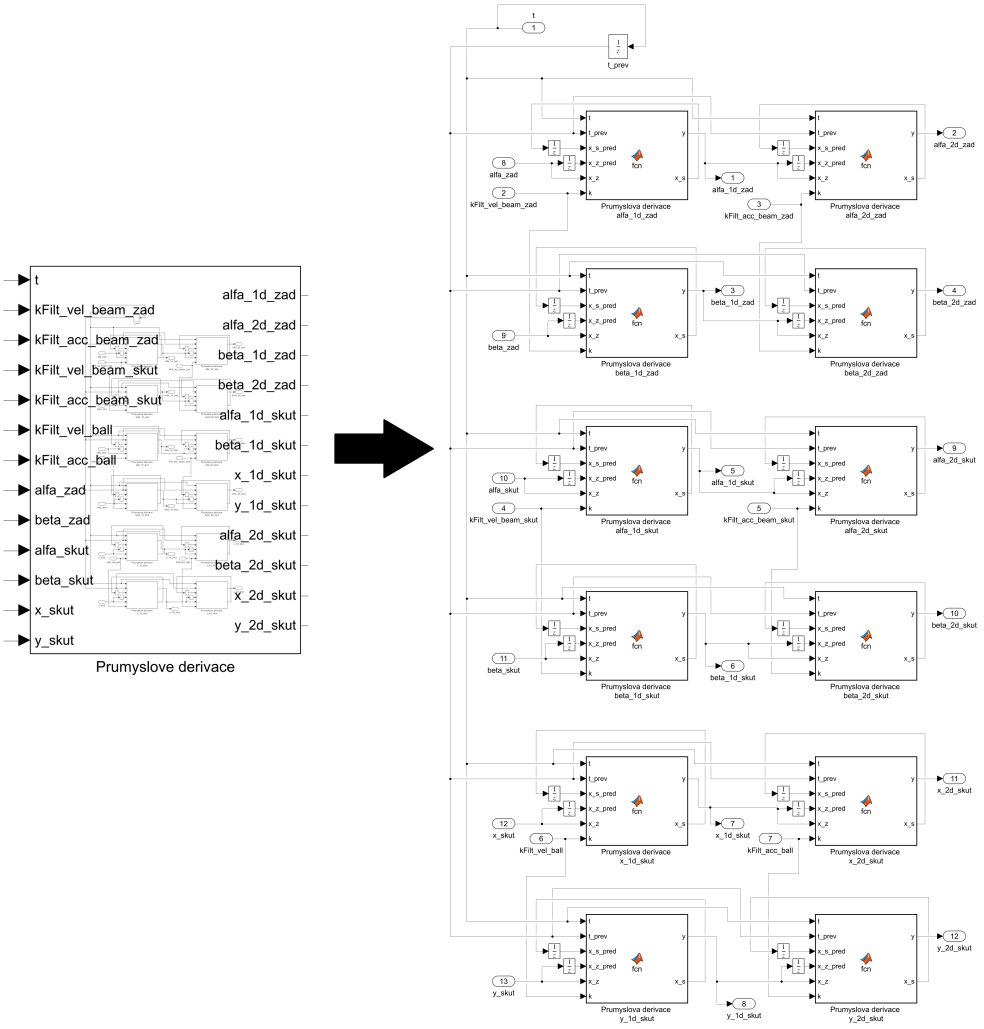
Obr. 4.96 Porovnání spojité, diskrétní a průmyslové/filtrované derivace - poloha

šuměný signál, jehož regulační odchylka se bude dále derivovat. První derivace polohové regulační odchylky, tedy rychlost, je zobrazena na Obr. 4.94, který obsahuje původní graf a 2 přiblížené průběhy.



Obr. 4.97 Porovnání spojité, diskrétní a průmyslové/filtrované derivace - zrychlení

Perioda vzorkování signálu je 1 ms a i při takto minimálně zašuměnému signálu



Obr. 4.98 Implementace průmyslové/filtrované derivace - SIMULINK

lze pozorovat znatelný rozdíl mezi běžnou numerickou a průmyslovou/filtrovanou derivací. Pokud se bude vstupní signál derivovat 2x, což je pro regulátor s výpočtem momentů nutné, bude výstupem zrychlení zobrazené na Obr. 4.97. V případě běžné numerické derivace se už jedná v podstatě o čistý šum (**modrý průběh**), průmyslová/filtrovaná derivace (**červený průběh**) však kopíruje ideální spojitou křivku zrychlení (**zelený průběh**) bez větších problémů. Takto rekonstruovaná derivace s nastavitelnou dynamiku přechodového děje je tedy

použitelná pro probíraný zákon řízení, a bude parametrizována podle možností regulovaného systému a požadavků na jeho dynamické chování. Průmyslová/filtrovaná derivace byla nejdříve namodelovaná a odladěna v MATLABu/SIMULINKu (Obr. 4.98) tak, aby ji poté bylo možné ve stejném smyslu přepsat do řídicího programu v C++ - Zdr. 29.

```
1 QPointF Regulation::PrumDer(float xZ, float xZ_prev, float xS_prev
      , float k)
2 {
3       pomPrumDer = xS_prev * (2 * k - Ta) / (2 * k + Ta) + Ta / (2
            * k + Ta) * (xZ_prev + xZ);
4       return QPointF(1 / k * (xZ - pomPrumDer), pomPrumDer);
5 }
```

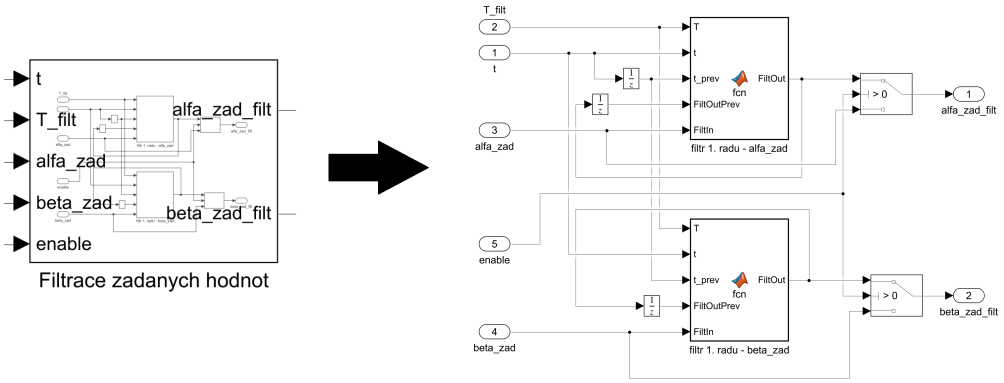
Zdr. 29 Zápis průmyslové/filtrované derivace

Zašumění snímaného signálu

Jak bylo již dříve uvedeno, na snímaný signál byl z důvodu co možná největšího přiblížení se realitě nasuperponován šum, v podobě harmonického průběhu kombinovaného s bílým šumem. Samotný šum je způsoben použitým snímačem otáček (enkodér kombinovaný s resolverem), přenosovou cestou a aplikovaným přepočtem. Harmonická nosná zase vychází ze suchého tření, z použitých materiálů a z vůlí - celek tvoří hysterezní smyčku. Výsledný šum je poté pouze přičtený ke vstupnímu signálu - lze vidět např. na Obr. 4.95.

Filtrace žádaných hodnot

Žádaná hodnota bude filtrovaná filtrem 1. řádu podle uvážení uživatele, s ohledem na charakter vstupního signálu. Filtr tedy musí být možné zapnout/vypnout, musí být nastavitelný a aplikovatelný pouze v některých případech a jeho SIMULINKové schéma je na Obr. 4.99.



Obr. 4.99 Implementace filtru žádaných hodnot - SIMULINK

Automaticky připínaná integrační složka

Z důvodu zmíněné vlečné chyby byla zavedena integrační složka, která se automaticky připíná na základě hodnoty regulační odchylky polohy/náklonu roviny. Pásmo připínání je parametrizovatelné, stejně jako časová konstanta integrace a volba, zda-li bude integrátor vůbec aplikovaný. Výstup integrátoru rovnou ovlivňuje žádaný proud a je začleněn až za hlavní regulační strukturu využívající výpočet momentů. Tím pádem musí být ošetřeno i přeintegrování způsobené paměťovým efektem integrace a saturace žádaného proudu na limit servozesilovače. Odpovídající zápis v kódu řídicího programu pro 1. DOF je ve Zdr. 30.

```
// pokud bude regulacni odchylka v zadane mezi, pripne se
    integracni slozka
if(fabs(rAngle1 - aAngle1) < TiCompTorqHyster*PI/180) {
     // pripinana I slozka regulatoru
     //*******************************************************
     Ki_rCurr1 = Ki_rCurr1Prev + 1 / TiCompTorq * (rAngle1 -
         aAngle1) * Ta;
     rCurr1 += Ki_rCurr1;
     // zastropovani integracni slozky prvniho kloubu
     if (rCurr1 < limitCurr1 && rCurr1 > -limitCurr1) {
          Ki_rCurr1Prev = Ki_rCurr1;
     }
} else Ki_rCurr1Prev = 0;
```

Zdr. 30 Zápis automaticky připínané integrační složky

Reálná perioda vzorkování a limity

Zásadní vliv na celý regulační pochod má skutečná perioda vzorkování regulátoru, resp. frekvence, s jakou je regulátor schopný přenastavovat žádaný proud. Regulace s výpočtem momentů má velice agresivní chování, které vyžaduje co možná největší vzorkovací frekvenci. Ta je ovšem na reálném systému z mnoha důvodů omezená, dostat se nad 1 kHz je obtížné a navíc není konstantní, protože řídicí počítač neobsahuje sytém reálného času. Pro přiblížení se realitě bude tedy perioda vzorkování omezena na 1 ms, což je nejnižší možná, ale ještě částečně zajistitelná hodnota, kterou lze považovat za konstantu. Při regulaci je však v drtivé většině případů volen režim maximální možné vzorkovací frekvence, kterou si řídicí jednotka sama volí podle vytíženosti vlákna a omezuje ji zejména rychlost komunikace mezi řídicím počítačem a servozesilovačem.

Všechny veličiny podléhající vzorkování jsou tedy i v simulačním modelu vzorkovány, a to podle periferie, které jsou podřízeny (poloha kuličky periodě vzorkování kamery, náklon roviny periodě vzorkování regulátoru). Taktéž jsou zavedeny ostatní limity, především momentový/proudový limit je důležitý. Ten je zohledňován i při nastavení dynamiky regulátoru - pozbývá totiž smysl nastavit regulátor na dynamiku, kterou není systém schopný zvládnout, a tím v podstatě vyřadit přímovazební část a přenechat celou regulaci na zpětnovazební části.

Regulátor s výpočtem momentů

Po provedeném předzpracování signálu a zajištění všech potřebných přizpůsobení, následuje samotný výpočet momentů (resp. proudů). Po přepsání obecných rovnic 4.58 a 4.59 do tvaru pro tento konkrétní systém s 2 řízenými DOF a zajištění schopnosti trackingu, bude vypadat zápis následovně:

$$\mathbf{D}\left(\alpha,\beta,x,y\right)\cdot\begin{bmatrix}v_1\\v_2\end{bmatrix}+\bar{H}\left(\alpha,\dot{\alpha},\beta,\dot{\beta},x,\dot{x},y,\dot{y}\right)+\bar{G}\left(\alpha,\beta,x,y\right)=\begin{bmatrix}Q_\alpha\\Q_\beta\end{bmatrix} \tag{4.62}$$

Pro ekvivalentní vstup $v=[v_1;\ v_2]^T$ platí:

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_{zad} \\ \ddot{\beta}_{zad} \end{bmatrix} + 2 \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \cdot \begin{bmatrix} \dot{\alpha}_{zad} - \dot{\alpha}_{skut} \\ \dot{\beta}_{zad} - \dot{\beta}_{skut} \end{bmatrix} + \begin{bmatrix} \lambda_1^2 \\ \lambda_2^2 \end{bmatrix} \cdot \begin{bmatrix} \alpha_{zad} - \alpha_{skut} \\ \beta_{zad} - \beta_{skut} \end{bmatrix} \tag{4.63}$$

kde $[\lambda_1;\ \lambda_2]^T > 0$ a její převrácená hodnota reprezentuje časovou konstantu konvergence regulační odchylky k nule. Jak je uvedeno v rovnici 4.62 a je patrné z názvu použité metody výpočtu regulátoru, výstupem je žádaný moment, který je dále nutné transformovat na žádaný proud. Do výpočtu tedy zasahuje ještě minimálně momentová konstanta, převodový poměr a účinnost použité převodovky, jak je uvedeno ve Zdr. 31, 32 a 33.

```
1 rCurr1 = ((0.2716 - 0.1716*bR - 0.02413*cos(2*beta_act) + 0.1119*
      bMassAct - 2.008e-5*sin(2.0*beta_act) + 3.179*pow(bR,2) +
      bMassAct*pow(y_act,2) - 0.04178*bR*cos(2*beta_act) + 9.44e-5*bR
      *sin(2.0*beta_act) + 0.669*bMassAct*y_act + 1.004*pow(bR,2)*cos
      (2*beta_act) + 1.4*pow(bR,2)*bMassAct) * (alfa_2d_des + 2*
      CompTorqLambdaAlfa*(alfa_1d_des - alfa_1d_act) + pow(
      CompTorqLambdaAlfa,2)*(alfa_des - alfa_act)) +
2 (0.000249*cos(beta_act) + 0.01142*sin(beta_act) - 0.5633*bR*sin(
      beta_act)) * (beta_2d_des + 2*CompTorqLambdaBeta*(beta_1d_des -
       beta_1d_act) + pow(CompTorqLambdaBeta,2)*(beta_des - beta_act)
      ) +
3 0.01142*pow(beta_1d_act,2)*cos(beta_act) - 0.000249*pow(
      beta_1d_act,2)*sin(beta_act) - 0.5633*bR*pow(beta_1d_act,2)*cos
      (beta_act) - 4.016e-5*alfa_1d_act*beta_1d_act*cos(2*beta_act) +
       0.04826*alfa_1d_act*beta_1d_act*sin(2*beta_act) + 0.669*
      alfa_1d_act*bMassAct*y_1d_act + 2*alfa_1d_act*bMassAct*y_act*
      y_1d_act - 2.008*alfa_1d_act*pow(bR,2)*beta_1d_act*sin(2*
      beta_act) + 0.0001888*alfa_1d_act*bR*beta_1d_act*cos(2*beta_act
      ) + 0.08356*alfa_1d_act*bR*beta_1d_act*sin(2*beta_act) +
4 0.4712*g*cos(alfa_act) + 0.06489*g*sin(alfa_act) + 0.04178*g*cos(
      beta_act)*sin(alfa_act) - 9.44e-5*g*sin(alfa_act)*sin(beta_act)
       + 0.3345*g*bMassAct*cos(alfa_act) - 2.175*bR*g*sin(alfa_act) -
       bR*g*bMassAct*sin(alfa_act) + g*bMassAct*y_act*cos(alfa_act) -
       2.008*bR*g*cos(beta_act)*sin(alfa_act)) /
5 (Km_1*rat_1*effRat_1);
```

Zdr. 31 Zápis regulátoru s výpočtem momentů - přesný mat model - 1. DOF

```
rCurr2 = ((0.000249*cos(beta_act) + 0.01142*sin(beta_act) -
    0.5633*bR*sin(beta_act)) * (alfa_2d_des + 2*CompTorqLambdaAlfa
    *(alfa_1d_des - alfa_1d_act) + pow(CompTorqLambdaAlfa,2)*(
    alfa_des - alfa_act)) +
(0.05051 - 0.08356*bR + 2.008*pow(bR,2) + bMassAct*pow(x_act,2) +
    1.4*pow(bR,2)*bMassAct) * (beta_2d_des + 2*CompTorqLambdaBeta*(
    beta_1d_des - beta_1d_act) + pow(CompTorqLambdaBeta,2)*(
    beta_des - beta_act)) +
2.008e-5*pow(alfa_1d_act,2)*cos(2*beta_act) - 0.02413*pow(
    alfa_1d_act,2)*sin(2*beta_act) + 1.004*pow(alfa_1d_act,2)*pow(
    bR,2)*sin(2*beta_act) - 9.44e-5*pow(alfa_1d_act,2)*bR*cos(2*
    beta_act) - 0.04178*pow(alfa_1d_act,2)*bR*sin(2*beta_act) + 2*
    beta_1d_act*bMassAct*x_act*x_1d_act +
9.44e-5*g*cos(alfa_act)*cos(beta_act) + 0.04178*g*cos(alfa_act)*
    sin(beta_act) - bR*g*bMassAct*sin(beta_act) + g*bMassAct*x_act*
    cos(beta_act) - 2.008*bR*g*cos(alfa_act)*sin(beta_act)) /
(Km_2*rat_2*effRat_2);
```

Zdr. 32 Zápis regulátoru s výpočtem momentů - přesný mat. model - 2. DOF

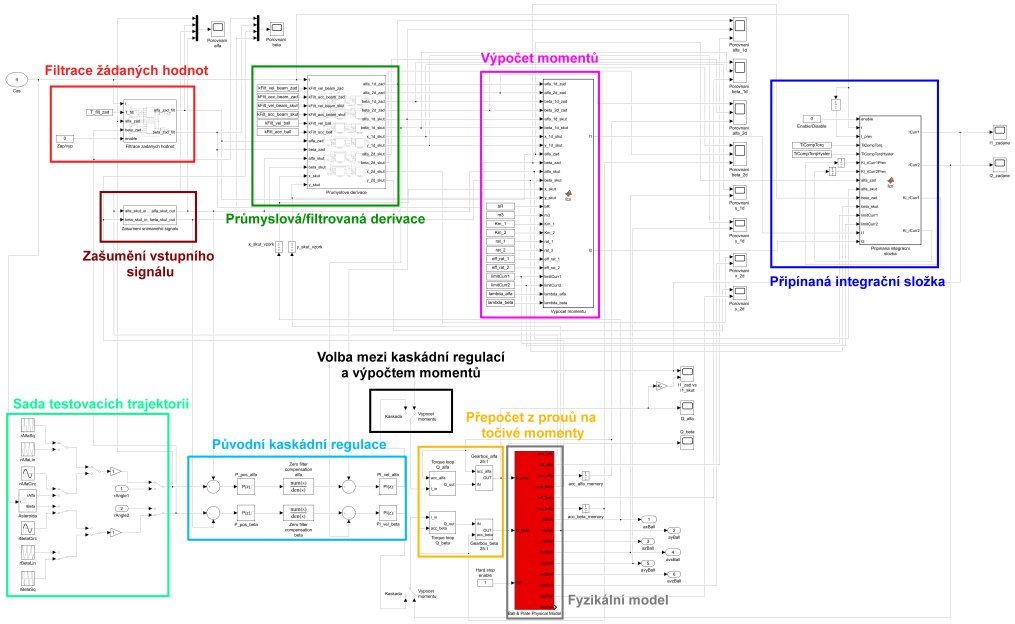
```
rCurr1 = (0.2716 * (alfa_2d_des + 2*CompTorqLambdaAlfa*(
    alfa_1d_des - alfa_1d_act) + pow(CompTorqLambdaAlfa,2)*(
    alfa_des - alfa_act)) +
0 +
0.4712*g*cos(alfa_act) + 0.3345*g*bMassAct*cos(alfa_act) + g*
    bMassAct*y_act*cos(alfa_act)) /
(Km_1*rat_1*effRat_1);

rCurr2 = (0.05051 * (beta_2d_des + 2*CompTorqLambdaBeta*(
    beta_1d_des - beta_1d_act) + pow(CompTorqLambdaBeta,2)*(
    beta_des - beta_act)) +
0 +
g*bMassAct*x_act*cos(beta_act)) /
(Km_2*rat_2*effRat_2);
```

Zdr. 33 Zápis regulátoru s výpočtem momentů - zjednodušený matematický model

Bylo zjištěno, že pro požadovanou dynamiku a pracovní rozsah pohybu není žádný pozorovatelný rozdíl, mezi využitím přesného a zjednodušeného matematického modelu. S důvodu výpočetní složitosti a přehlednosti

bude tedy dále uvažován zjednodušený matematický model, uvedený ve Zdr. 33 a rovnicích 4.35 a 4.36.

Blokové schéma celé regulační struktury s výpočtem momentů je uvedeno na Obr. 4.100, včetně vyznačení probíraných částí předzpracování signálu a původní kaskádní regulace, která bude sloužit pro porovnání a vyhodnocení dat.



Obr. 4.100 Kompletní simulační model regulátoru s výpočtem momentů - SIMULINK

Pro naladění regulátoru a závěrečné testy kvality regulace byly vytvořeno 5 typových křivek, které svým charakterem a dynamikou pohybu pokrývají širokou škálu možných požadavků na řízení a jsou plně parametrizovatelné. Jedná se o tyto trajektorie:

- **Statická** - žádanou hodnotou je konstantní náklon roviny o úhlech α a β, který se v čase nemění. Ověřuje se schopnost regulátoru stabilizovat rovinu v přesně dané pozici v reakci na poruchu, kdy budou všechny derivace stavových proměnných v ustáleném stavu nulové.
- **Skoky** - podobně jako u předchozího případu je žádanou hodnotou konstantní náklon roviny o úhlech α a β, která se však v čase skokově mění ve

4 různých konfiguracích. Ověřuje se, jak rychle dokáže regulátor stabilizovat rovinu v reakci na skok, a to v celém rozsahu náklonu roviny. Obdobně jako pro statický náklon jsou v ustáleném stavu všechny derivace stavových proměnných nulové.

- **Lineární** - předchozí případ se skokovou změnou žádané hodnoty je upraven do podoby, kdy se vrcholy spojily navzorkovanou přímkou a právě tato přímka vstupuje do regulace jako žádaný úhel náklonu jednotlivých os. Lineární změna žádané hodnoty už klade požadavky na její přesné sledování, ustálený stav již obsahuje konstantní první derivaci stavových proměnných a nulovou druhou derivaci (kromě změny sklonu přímky). Testuje se schopnost trackingu, možnost nahrazení skokových změn žádané hodnoty rampou a vliv automaticky připínané integrační složky polohové smyčky.

- **Kružnice** - žádanou hodnotou je trajektorie opisující kružnici, tedy harmonicky se měnící úhly náklonu roviny v čase tak, aby po jejich vzájemně závislém vynesení do grafu vznikla kružnice. Zde se zjišťuje, jak reaguje řídicí systém na harmonickou změnu žádané hodnoty pro různé frekvence a do jaké míry je schopný ji sledovat. V ustáleném stavu se všechny stavové proměnné i jejich derivace harmonicky mění.

- **Asteroida** - reprezentuje maximální dynamiku změny všech stavových parametrů. Je speciální případ hypocykloidy se 4 vrcholy, u které se žádaný náklon roviny periodicky opakuje v křivkách, složených z více harmonických signálů. Tím je zajištěna dynamická neharmonická změna všech stavových proměnných. Testuje se schopnost trackingu systému v závislosti na dynamických neharmonických změnách žádané hodnoty.

Na výše uvedených trajektoriích bude také probíhat porovnání s kaskádní regulací, která je zde považována za jakýsi etalon pro řízení pohybu.

Nastavení regulátoru s výpočtem momentů

Metod pro nastavení tohoto typu regulátoru je mnoho, od čistě experimentální [55], přes využití Ziegler-Nicholsovy metody [60], až po samočinné nastavení neuronovou sítí [28]. Ve všech případech ale pobíhá ladění pouze zpětnovazebních zesílení PD regulátoru, v tomto případě je stěžejní částí nastavení časových konstant průmyslových/filtrovaných derivací všech potřebných signálů, filtru žádaných hodnot, integračního času automaticky připínaného integrátoru, a to vše s ohledem na reálně dosažitelnou dynamiku systému, omezenou periodu vzorkování regulátoru, proudového/momentového limitu, možný vznik proudové rezonance atd. Paralelně se musí porovnávat výsledky simulace s reálným systémem, což vylučuje většinu možných přístupů a metoda pokus-omyl také selhává. Proto je potřeba navrhnout systematický přístup, který využívá znalosti řízeného systému jako celku, a postupně konfiguruje celý regulační proces.

Níže popsaná metoda je navržena autorem práce s ohledem na výše uvedené okolnosti a skládá se z několika bodů s pevně danou posloupností. Pro začátek je vhodné definovat parametry, které jsou nastavitelné a jejich charakter odpovídá v podstatě konstantám regulátoru v běžné PID regulační struktuře. Jsou jimi:

- $Ti_{FiltDes}$ - časová konstanta filtrace žádaných hodnot
- $En_{FiltDes}$ - povolení/zakázaní filtrace žádaných hodnot
- $Ti_{DerDesVelBeam}$ - časová konstanta průmyslové/filtrované derivace žádané hodnoty rychlosti nakloněné roviny
- $Ti_{DerDesAccBeam}$ - časová konstanta průmyslové/filtrované derivace žádané hodnoty zrychlení nakloněné roviny
- $Ti_{DerActVelBeam}$ - časová konstanta průmyslové/filtrované derivace skutečné hodnoty rychlosti nakloněné roviny
- $Ti_{DerActVelBall}$ - časová konstanta průmyslové/filtrované derivace skutečné hodnoty rychlosti kuličky
- λ_α - zpětnovazební zesílení pro 1. DOF
- λ_β - zpětnovazební zesílení pro 2. DOF

- Ti_{Int} - časová konstanta automaticky připínané integrační složky
- $Hyst_{Int}$ - pásmo hystereze, ve kterém je aktivní integrátor
- En_{Int} - povolení/zakázaní připínání integrační složky

Všechny parametry jsou nastavitelné jak v simulaci, tak v reálném systému. Pro přehlednost a vměstnání všech parametrů do aplikace byla záložka *Regulace* ještě rozdělena do dvou pod-záložek, jedna pro regulaci kuličky, druhá pro regulaci roviny - viz náhled na Obr. 4.101. Je zde také demonstrována úprava záložky *Kalibrace*, která byla rozšířena o možnost ukládání naměřených dat pro kontrolu nastavovaného regulátoru a další statistické vyhodnocení. Při měření náklonu roviny je možná nastavit požadovanou periodu vzorkování, typ testovací trajektorie, její amplitudu a periodu opakování, počet požadovaných měření a délku záznamu. Ukládají se skutečné úhly náklonu roviny, žádané proudy, skutečné proudy a reálný vzorkovací čas.

Obr. 4.101 Náhled na parametrizaci regulátorů a záznam dat v GUI

Při nastavování regulátoru s výpočtem momentů se nejdříve zvolí testovací trajektorie, na které bude probíhat ladění. V tomto případě se bude jednat o lineární průběh žádaných úhlů náklonů, který začíná v ustálené pozici - tento typ trajektorie se ze všech uvedených nejvíce osvědčil. Lineární změnou žádané polohy

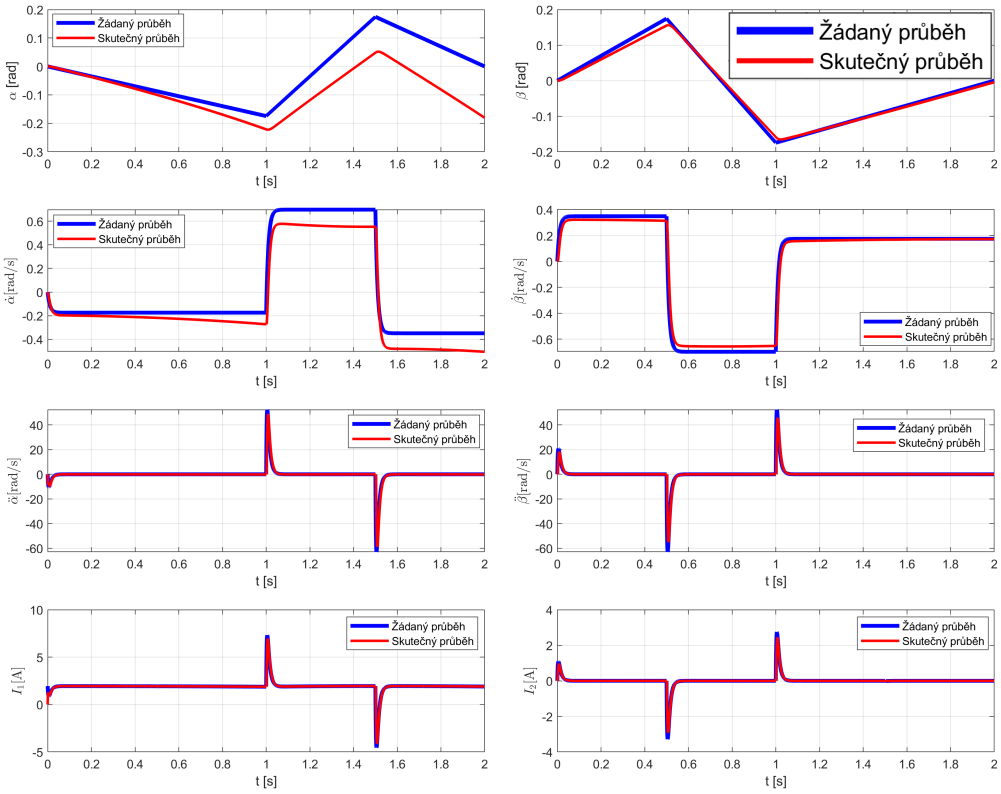
bude zajištěn po průchodu průmyslovou derivací filtrovaný skok v rychlosti a filtrovaný impulz ve zrychlení, což jsou fyzikálně splnitelné požadavky. Dále lze na této trajektorii kontrolovat proudové rázy (potažmo proudovou rezonanci) při změně sklonu přímky, fázový posuv, schopnost trackingu, vlečnou chybu a oscilace. Po výběru a nastavení dynamiky testovacích průběhů bude pro ladění regulátoru použitý následující postup:

- Vynulují se obě zesílení λ_α a λ_β, tím se vyřadí zpětnovazební část regulátoru. Poté bude možné nastavit přímovazební část tak, aby skutečné hodnoty co možná nejpřesněji kopírovaly požadovaný přechodový děj.

- Vyřadí se šum, filtrace žádaných hodnot i připínání integrační složky. To jsou pouze doplňkové části regulace sloužící k vyhlazení a doladění regulačního pochodu, nesmí být do základního nastavení regulátoru zahrnuté.

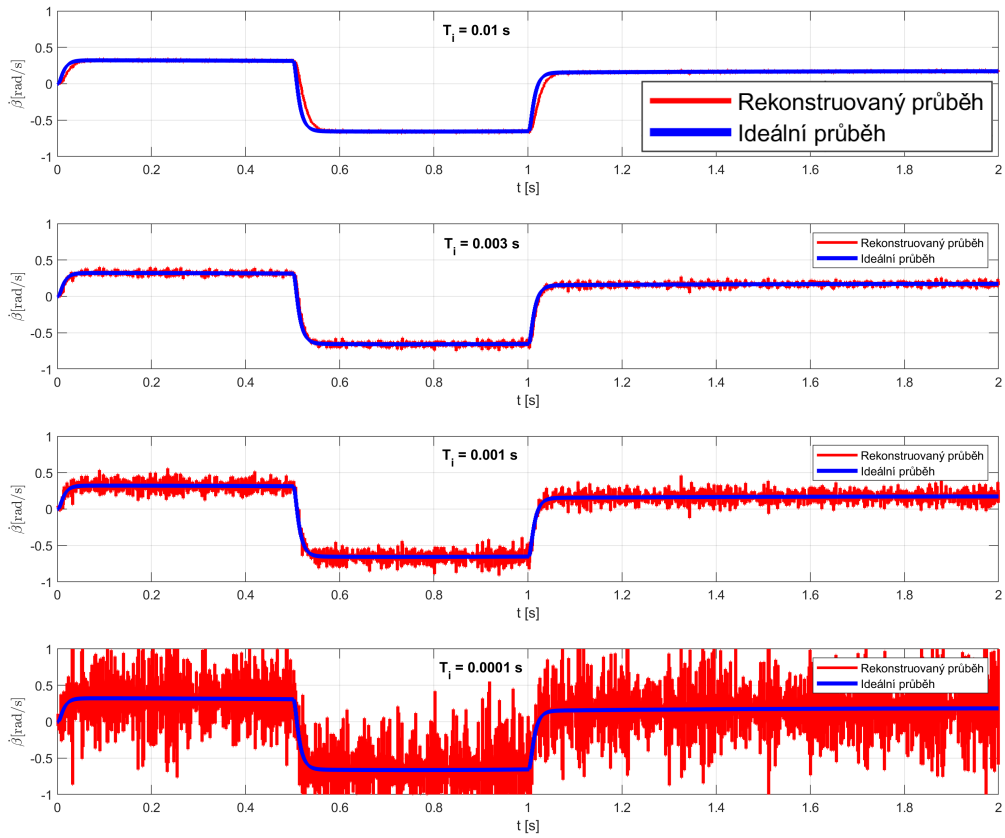
- Nastaví se časové konstanty průmyslové/filtrované derivace žádaných hodnot $Ti_{DerDesVelBeam}$ a $Ti_{DerDesAccBeam}$ tak, aby jejich přechodový děj pokud možno kopíroval průběhy odpovídajících skutečných hodnot. Začíná se na hodnotách podle určené/odhadované dynamiky přechodového děje daného průběhu. Zároveň se kontroluje průběh rychlosti i zrychlení, časová konstanta $Ti_{DerActVelBeam}$ musí být nastavena na nízkou hodnotu (např. 100 µs), aby neměnila amplitudu ani fázi porovnávacího signálu. V této chvíli by nastala proudová rezonance pouze pro velmi špatné nastavení, ale pro jistotu se kontroluje také průběh žádaného proudu s cílem pokud možno ani v jednom kloubu nedosáhnout saturace. Tím se ponechá rezerva pro zpětnovazební část i na začátek přechodového děje. Příklad nastavené přímovazební větve je na Obr. 4.102 a odpovídá hodnotám $Ti_{DerDesVelBeam} = 0.01s$ a $Ti_{DerDesAccBeam} = 0.003s$.

- Dalším krokem je nastavení časové konstanty $Ti_{DerActVelBeam}$ tak, aby se rekonstruovaná úhlová rychlost co nejméně zpožďovala za snímanou rychlostí a zároveň se eliminovaly šumy. Zde se musí zvolit vždy kompromis, protože pokud je vstupní signál zašuměný, a to je v reálném systému vždy, velká časová konstanta zajistí dobré odfiltrování šumu, ovšem za cenu fá-



Obr. 4.102 Nastavení regulátoru s výpočtem momentů - přímovazební část

zového posunutí a zkreslení průběhu rekonstruované veličiny. Malá časová konstanta bude zase zvyšovat šumy, ale fázový posuv a zkreslení bude minimální (pokud se za zkreslení samozřejmě nepovažuje šum). Derivace snímané veličiny se uplatňuje mimo matici H (z pohybových rovnic) hlavně ve zpětné vazbě, a to pouze první derivace. Druhá derivace (viz rovnice 4.63) je potřebná pouze z žádané hodnoty. Nyní se tedy povolí šum ve snímané veličině a vhodně se nastaví časová konstanta $Ti_{DerActVelBeam}$. Protože bude mít její nastavení vliv na zašumění žádaného proudu, který bude zvyšován narůstajícím zpětnovazebním zesílením, toto nastavení nemusí být konečné a bude se v průběhu ladění pravděpodobně upravovat. Různé nastavení s odpovídajícím zašuměním rekonstruovaného signálu (úhlová rychlost v 2. DOF) jsou zobrazeny na Obr. 4.103.

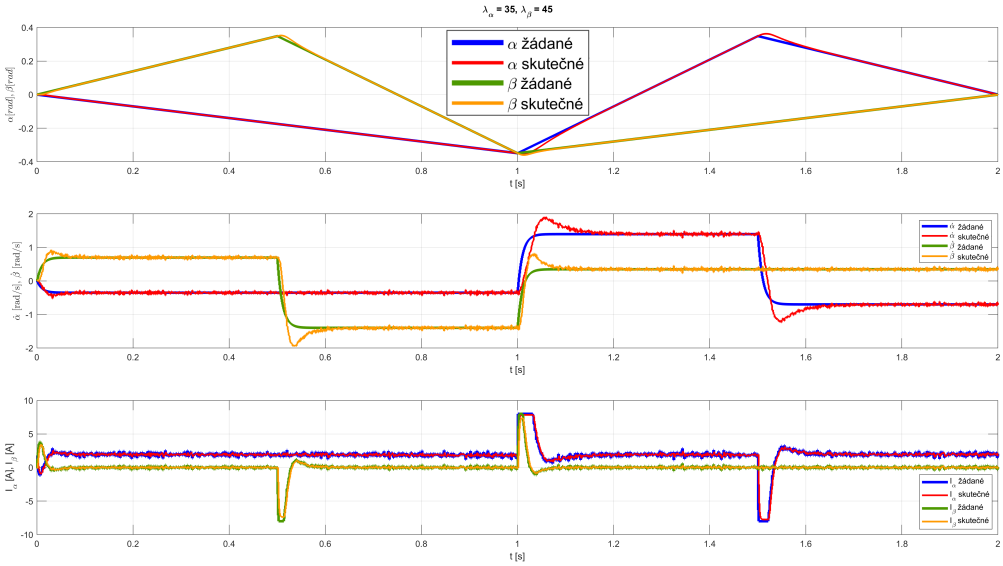


Obr. 4.103 Nastavení regulátoru s výpočtem momentů - rekonstrukce skutečné úhlové rychlosti 2. DOF

Jak vidno, pro časovou konstantu $T_i = 0.01s$ je signál velice dobře odfiltrovaný od šumů, ale dochází ke zpoždění, což by mělo negativní vliv při rychlých změnách žádaných hodnot a vznikla by vlečná chyba. Poslední z grafů je zase naopak velice zašuměný a použití takovéhoto nastavení by vedlo k rozkmitání systému a k proudové rezonanci. Průběhy pro $T_i = 0.003s$ a $T_i = 0.001s$ jsou již dobrým kompromisem mezi rychlostí reakce na změnu vstupní veličiny a filtrováním šumu. Časová konstanta v tomto rozmezí je použitelná a bude doladěna při finálním nastavení. Pro další kroky bude zvolena hodnota $T_i = 0.003s$.

- Nyní nastává čas pro zařazení zpětnovazební části regulátoru tak, že se za-

čnou zvyšovat hodnoty zpětnovazebních zesílení λ_α a λ_β. Zde už by se měl skutečný průběh přibližovat žádané hodnotě, sledují se tedy úhly náklonu a kontroluje se průběh proudů. Malé hodnoty konstant λ_α a λ_β způsobují pomalé přibližování se žádané hodnotě, velkou trvalou regulační odchylku a malé proudové rázy. Vysoké hodnoty zpětnovazebních zesílení zase zesilují vliv zašuměné snímané hodnoty, zapříčiňují velké proudové rázy až proudovou rezonanci, ale zajišťují rychlý regulační pochod a malou vlečnou chybu. Je nutná volba kompromisu vzhledem k charakteru žádané hodnoty, konečné nastavení v tomto případě odpovídá $\lambda_\alpha = 35$ a $\lambda_\beta = 45$ - průběhy na Obr. 4.104. Výsledky pro malé a velké hodnoty zpětnovazebních zesílení jsou uvedeny v PŘÍLOZE M.



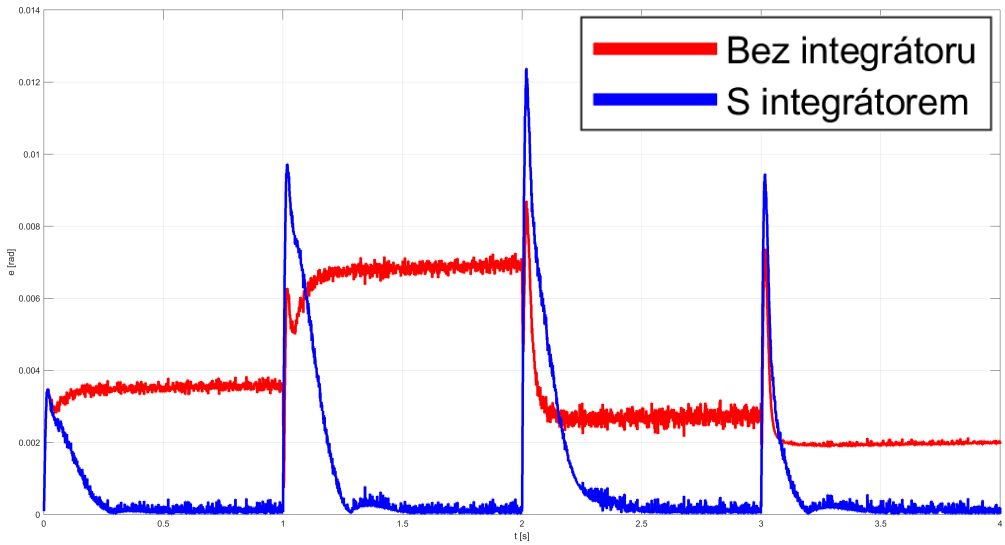
Obr. 4.104 Nastavení regulátoru s výpočtem momentů - $\lambda_\alpha = 35$ a $\lambda_\beta = 45$

- Dalším krokem je zařazení možnosti filtrace žádané hodnoty. Zejména při jejich rychle se měnících skokových změnách dochází k velkým proudovým rázům, což vychází ze samotného principu metody výpočtu momentů. Filtr obsažený v průmyslové/filtrované derivaci je aplikovaný pouze na rychlost a zrychlení, skoková změna v poloze není stále nijak omezená a vy-

soké zpětnovazební zesílení zapříčiní okamžitou proudovou/momentovou špičku. Pozvolná změna při požadavku na skok bude zajištěna filtrem (1. řádu) žádané hodnoty. Tento filtr samozřejmě posouvá fázi, a proto je vhodné jej aplikovat zejména na rychle se měnící skokové změny, kde by při jeho absenci došlo k proudové rezonanci, nebo na omezení dynamiky počáteční fáze regulačního pochodu. U lineárního/harmonického/neharmonického průběhu by byla posunutá fáze již dosti patrná z vlečné chyby (PŘÍLOHA N.b). Časovou konstantu filtru je potřeba udržet na co nejnižší přijatelné hodnotě, která byla dle měření a testů, zejména na reálném systému, stanovena na $Ti_{FiltDes} = 0.09s$. Bez použití filtru žádaných hodnot úhlů náklonů roviny při regulaci kuličky je metoda výpočtu momentů v podstatě nepoužitelná.

- Poslední volitelnou částí regulátoru je automatické připínání integrační složky ve zvoleném aktivním pásmu. Oproti filtraci žádaných hodnot je integrační složka naopak vhodná u všech průběhů kromě skokových změn, zejména těch rychlých (zde by docházelo ke zbytečnému rozkmitání systému). U lineárního/harmonického/neharmonického průběhu zajistí integrační složka kompenzaci nepřesností matematického modelu, a tedy teoreticky nulovou vlečnou chybu. Rozdíly jsou opět pozorovatelné hlavně na reálném systému, který obsahuje oproti fyzikálnímu modelu pochopitelně mnohem více nepřesností/poruch - viz. Obr. 4.114 a PŘÍLOHA N.a. Při detailním přiblížení lze vidět rozdíl ve vlečné chybě i na fyzikálním modelu. Příklad vykreslení Euklidovské odchylky pro požadovaný lineární průběh s integrační složkou a bez ní, je zobrazený na Obr. 4.105. Konečné nastavení časové konstanty integrátoru a hystereze odpovídá $Ti_{Int} = 0.001s$ a $Hyst_{Int} = 3$.

Regulátor s výpočtem momentů je tedy nastaven a bude dále porovnáván na typových trajektoriích s kaskádní regulací. Nastavování regulace probíhalo vždy primárně na fyzikálním modelu, až po odladění bylo aplikováno na reálný systém. Zde byly po testech na typových trajektoriích upraveny parametry do konečné podoby tak, aby regulace fungovala pro všechny typové průběhy beze změn konstant regulátoru.



Obr. 4.105 Průběh Euklidovské odchylky regulátoru s výpočtem momentů pro lineární trajektorii

4.8.6 Další možné metody řízení pohybu

V této podkapitole je uvedeno několik dalších možných způsobů řízení pohybu, u kterých byla provedena pouze teoretická rozvaha nad možnostmi jejich použití. Z daleka neobsahuje všechny dostupné metody, ale pouze některé autorem vybrané, které jsou analyzovány z pohledu aplikovatelnosti na tento konkrétní systém.

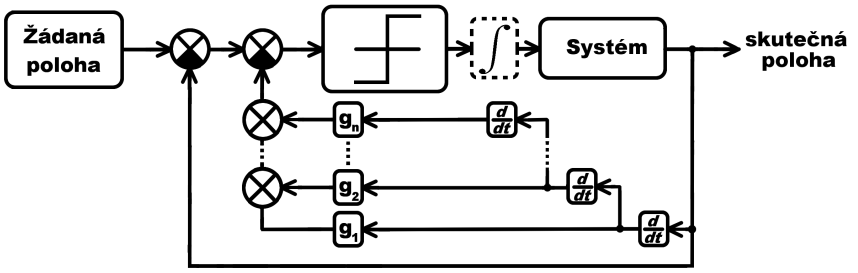
Řízení v klouzavém režimu (sliding mode)

Tento typ řízení (stejně jako metoda výpočtů momentů) patří také mezi algoritmy regulace nelineárních systémů a je formou dvoupolohového řízení. Návrh a realizace tohoto regulátoru je velmi jednoduchý, je potřeba znát pouze relativní řád systému. Použitý algoritmus využívá vysoké frekvence přepínání okolo tzv. přepínací funkce. Relativní řád uzavřeného regulačního obvodu musí být vždy roven jedné, proto je nutné při regulaci systému s vyšším řádem zařadit do zpětné vazby derivační filtry, které relativní řád snižují. Celý řídicí systém je extrémně robustní, protože přenosová funkce nezávisí na parametrech systému a na externích poruchách. Vysoká frekvence spínání okolo přepínací funkce umožňuje

také dobré potlačení případných chyb. Při použití v mechanických systémech nemusí být ovšem vysokofrekvenční kmitání vždy vhodné, jelikož může docházet k opotřebení různých součástí a může také dojít k vybuzení vlastní dynamiky systému.

Čistý „klouzavý“ regulátor má jen dvouhodnotový výstup, výsledná trajektorie stavového bodu proto nesleduje přepínací křivku přesně, ale pohybuje se v jejím blízkém okolí - také je uváděno, že za ní „klouže“, odtud tento termín. Pokud by byl stavový bod udržovaný stále na přepínacím rozhraní, je zpětnovazební dynamika dána pouze rozhraním samotným a je tedy nezávislá na parametrech systému a na externích poruchách.

V reálné aplikaci se ovšem často žádá ustálení na žádané hodnotě, eliminace přepínání řízení v ustáleném stavu, omezení velikosti akčního zásahu, aproximace reálně nerealizovatelné přepínací funkce signum apod. Při ošetření těchto požadavků se do systému v případě ustálení na žádané hodnotě zavádí regulátor s integrační složkou, při eliminaci přepínání v ustáleném stavu se používá buď aproximace přepínací funkce, nebo se zavádí vyhlazovací integrátor. V případě omezení akčního zásahu se aplikuje softwarový přepínač, který přepíná mezi klouzavým režimem a náhradní hodnotou. Při každé změně zasahující do výsledného regulačního obvodu je potřeba opět snížit relativní řád uzavřeného regulačního obvodu na jedničku, pokud se např. při zavedení integrátoru řád zvýšil.



Obr. 4.106 Zjednodušený blokový diagram pro řízení v klouzavém režimu

Komplikací při použití této metody na probíraný systém by byla bezpochyby vysoká frekvence spínání, která by po čase zvýšila vůli v mechanické soustavě a rozkmitala pružné části systému (za předpokladu, že by tato vysoká frekvence

spínání byla vůbec řídicím počítačem dosažitelná). Systém by byl v podstatě permanentně v proudové rezonanci, která je vzhledem k vůlím, tření a pružnostem modelu nežádoucí. Stěžejním bodem pro správné nastavení regulace je také určení proporcionálních zesílení u zpětných vazeb - každý derivační člen zavádí do regulované soustavy šum, což je další bod k řešení (v regulátoru s výpočtem momentů řešeno průmyslovou/filtrovanou derivací). Vše výše zmíněné je pozorovatelné zejména na reálném systému, ve fyzikálním modelu pouze omezeně, a tak by bylo i naladění regulátoru pro fungování v reálných podmínkách komplikované. Zásadní nelinearita v navrženém systému vzniká až při vzájemné interakci mezi jednotlivými stupni volnosti, proto je otázkou, jestli by přinesl tento typ řízení pozorovatelné vylepšení regulačního procesu, pokud by byl vůbec realizovatelný [55, 79, 1].

Přímá Ljapunovova metoda

Obecný problém stability pohybu uvádí dvě metody analýzy stability - **metoda linearizace** a **přímá metoda**. Metoda linearizace řeší souvislosti mezi lokální stabilitou nelineárního systému v okolí rovnovážného bodu a stabilitou jeho lineární aproximace. Přímá metoda naproti tomu není omezena pouze na lokální pohyb, ale určuje vlastnosti stability nelineárního systému pomocí skalární „energetické“ funkce systému a vyšetřováním jejich časových změn. Základní myšlenka této metody je:

„*Jestliže je celková energie mechanického/elektrického systému trvale disipována (přeměňována na teplo), pak systém, ať už lineární nebo nelineární, musí nakonec přejít do svého rovnovážného stavu.*“[80]

Existuje mnoho vět, důkazů a postupů při analýze systému. Všechny jsou však založené na znalosti explicitní Ljapunovovy funkce. Doposud ale nebyl objeven obecný efektivní postup jejího nalezení pro nelineární systémy. Po pochopení fyzikálního významu a na základě zkušenosti a intuice, lze však použít několik postupů zjednodušujících tento komplikovaný a nepřehledný způsob nalezení Ljapunovovy funkce. Mezi tyto metody patří např. Krasovského metoda, ověřující, zda specifický výběr Ljapunovovy funkce opravdu vede k řešení, nebo Metoda gradientu, zahrnující předpoklad určitého tvaru gradientu, jeho integrace a

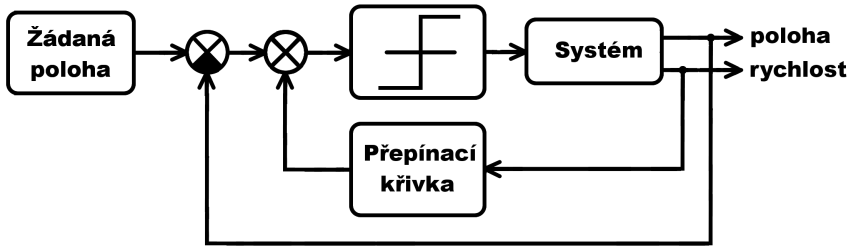
následného hledání Ljapunovovy funkce. Tyto metody vedou občas k úspěšnému nalezení Ljapunovovy funkce, ale to zejména pro systémy nízkých řádů. U robotických systému s více stupni volnosti je použití této metody velice komplikované a vyžaduje detailní znalost fyzikálních vlastností řízeného systému.

Přímá Ljapunovova metoda není však použitelná pouze pro analýzu stability systému, ale také pro návrh jeho řízení. Existují v podstatě dva přístupy k použití přímé Ljapunovovy metody, obě mají nicméně co do činění s metodou pokus-omyl. První technika předpokládá určitou formu zákona řízení a poté hledá Ljapunovovu funkci pro ověření správného výběru. Druhá technika naopak požaduje výběr určité Ljapunovovy funkce a posléze hledá zákon řízení, který tuto funkci realizuje.

Použití přímé Ljapunovovy metody je obecně velice komplikované a pro systémy vyšších řádů v podstatě vyloučené. Všechny komplikace vyplývají ze složitosti nalezení Ljapunovovy funkce, tedy skalární funkce popisující „energetický" stav systému. Její reálná aplikovatelnost na probíraném systému se nepředpokládá, není však vyloučená [80].

Pontrjaginův princip maxima

Pontrjaginův princip maxima určuje nutnou podmínku pro existenci zákona řízení optimalizujícího daný kriteriální funkcionál, během „pohybu" systému z výchozího stavu do stavu koncového, při určené množině přípustných vstupů. Tj. existuje-li optimální přípustné řízení (vstup) systému popsaného tzv. kanonickým tvarem obyčejných diferenciálních rovnic (stavový popis), pak Pontrjaginův princip říká, že optimální řízení bude to, které maximalizuje Hamiltonian podél celé trajektorie pohybu. Současně uvádí vztah mezi popisem dynamiky systému a novým vektorem funkcí vystupujících v Hamiltonianu, který umožňuje určit optimální vstup.
Cílem optimálního řízení je tedy nalezení trajektorie řízené stavové proměnné a řídicího vstupu převádějící systém z počátečního stavu do koncového stavu, za podmínky minimalizace kriteriálního funkcionálu (např. minimalizace dodané energie, času přechodu, vzdálenosti apod.) - rovnice 4.64.



Obr. 4.107 Zjednodušený blokový diagram pro časově optimální řízení soustavy 2. řádu s nekmitavými póly při aplikaci Pontr. principu maxima

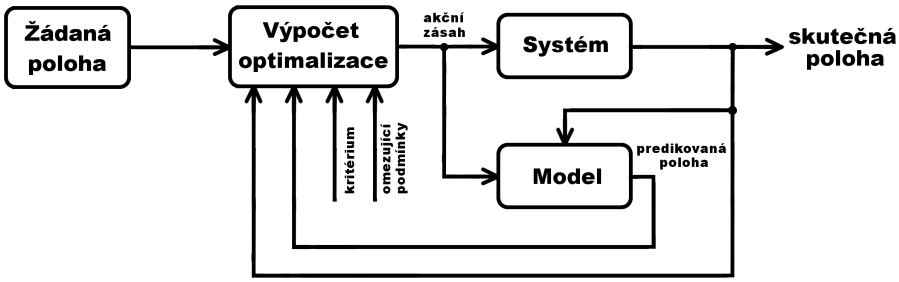
$$J = \int_{t_0}^{t_f} f_0 \left[\bar{x}\left(t\right), \bar{U}\left(t\right) \right] \cdot dt \tag{4.64}$$

Pokud je na této množině přípustných řešení optimální řízení převádějící soustavu z počátečního stavu do stavu koncového po optimální trajektorii, tedy minimalizující rovnici 4.64, pak existuje nenulová vektorová funkce, při níž má zavedená Pontrjaginova (Hamiltonova) funkce v libovolném čase $t_0 \leq t \leq t_f$ maximální hodnotu.

Je-li známé optimální řízení jen jako funkci času, pak je toto řízení použitelné pouze v otevřené smyčce. V praxi je však zájem zjistit tzv. optimální zákon řízení umožňující použití zpětné vazby, tedy popis vstupní veličiny jako funkce stavu systému. Vlivem nelinearity pohybových rovnic robotických struktur neexistuje analytické řešení a musí být použito řešení numerické. V probíraném případě není předem známá žádaná poloha, proto je nutné numerický výpočet vykonávat v reálném čase. Reálné použití této metody se tedy vzhledem ke složitosti řešeného systému opět nepředpokládá, pokud by však bylo realizovatelné, pak pravděpodobně pouze pro první-druhý stupeň volnosti, které mají jednoznačnou kinematickou vazbu [79].

Rychlé online prediktivní řízení

Prediktivní řízení je moderní metodou regulace, založenou na znalosti matematického modelu řízeného objektu a možnosti predikce jeho chování. K tomu, aby bylo možné tuto metodu implementovat, je nutné znát matematický model řízeného systému, budoucí hodnotu referenčního signálu a časový horizont predikce. Cílem je poté výpočet řídicího signálu (akčního zásahu), který zaručí budoucí chování řízeného systému podle známého referenčního signálu. Toho je docíleno minimalizací účelové funkce, která minimalizuje odchylku mezi žádanou a výstupní hodnotou. Výsledkem je vektor budoucích hodnot akčního zásahu, z něhož se použije pro řízení pouze první hodnota a celý výpočet se v další periodě vzorkování opakuje. Z tohoto faktu vyplývá první úskalí, kterým je náročnost na hardware.



Obr. 4.108 Zjednodušený blokový diagram pro prediktivní řízení

Pro řízení nelineárních systémů je možné využít tzv. síť lokálních modelů. Tato metoda je založena na linearizaci systému okolo několika zvolených pracovních bodů, mezi kterými se bude regulace na základě aktuálního stavu přepínat (linearizace je závislá na míře nelinearity řízeného systému a bude přesná jen v blízkém okolí zvoleného pracovního bodu). Síť lokálních lineárních modelů je vhodná pro případy, kdy je potřeba řídit systém v celém jeho rozsahu, ne pouze v určitém okolí zvoleného pracovního bodu.

Je několik způsobů, jak mezi modely přepínat. První možností je přepínat mezi odvozenými modely přímo, což způsobí ostré přechody a může mít výrazný vliv na výslednou stabilitu systému. Druhý způsob je lineární přechod, kdy je výsledný model kombinací dvou sousedních modelů. Třetím způsobem je neli-

neární přechod mezi modely, kdy je výsledný model vypočítáván pomocí váhové funkce [72, 16, 25].

Hlavními možnými komplikacemi při použití této metody by mohl být předpoklad znalosti budoucích žádaných hodnot a výpočetní nároky celé metody. Není jisté, jestli by řídicí počítač zvládal vykonávat celý optimalizační výpočet v každé periodě vzorkování a neprodlužoval ji tak za únosnou mez. Znalost budoucích hodnot referenčního signálu by mohl být také problém, to ale záleží na úrovni zařazení prediktivního regulátoru do regulačního obvodu. Složitost matematického popisu by také hrála v optimalizačním výpočtu roli; jak velkou, to by záleželo až na samotné implementaci.

4.8.7 Porovnání regulátorů náklonu roviny

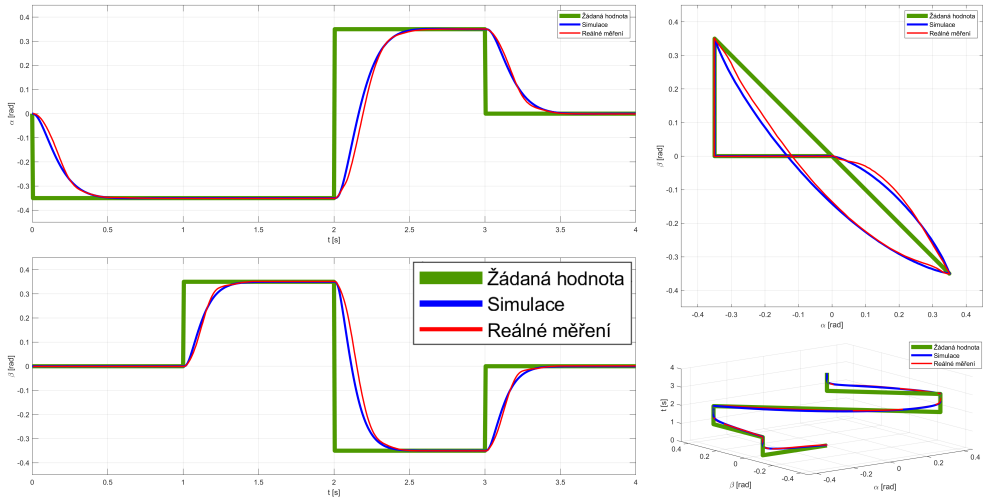
Jako představitel nejrozšířenější a běžně používané metody řízení pohybu byla vybrána **kaskádní regulace** (dále KR), za moderní metodu řízení pohybových stavů mechanické soustavy byla zvolena metoda regulace pomocí **výpočtů momentů** (dále RVM). Tyto dva přístupy byly porovnány na vybraných typových trajektoriích popsaných v podkapitole 4.8.5. Důraz byl kladen zejména na sledování žádané hodnoty (schopnost trackingu), což je u robotických soustav stěžejní požadavek. Každá charakteristika zobrazující chování reálného systému je výsledkem statistického vyhodnocení z 10 nezávislých měření, z kterých je spočítán medián. Tento výsledek je poté porovnán s výsledkem měření na fyzikálním modelu a s žádanou hodnotou. Grafy zobrazují časové závislosti obou úhlů náklonu, vzájemnou závislost jednoho úhlu na druhém a 3D křivku této vzájemné závislosti na čase. Amplituda průběhu je vždy volena na 20° náklonu roviny pro oba úhly, perioda je pro porovnání při různé dynamice testována na hodnotách 1 s, 2 s a 4 s.

Skoková změna žádané hodnoty

První porovnání probíhalo na skokové změně žádané hodnoty. Zde lze jednoznačně tvrdit, že u RK bylo dosaženo lepších výsledků, než u RVM. Při porovnávání průběhů pouze z fyzikálního modelu by tomu sice bylo naopak, za objekt zájmu je však považován reálný systém, fyzikální model slouží pouze pro analýzu

systému a ladění regulátoru.

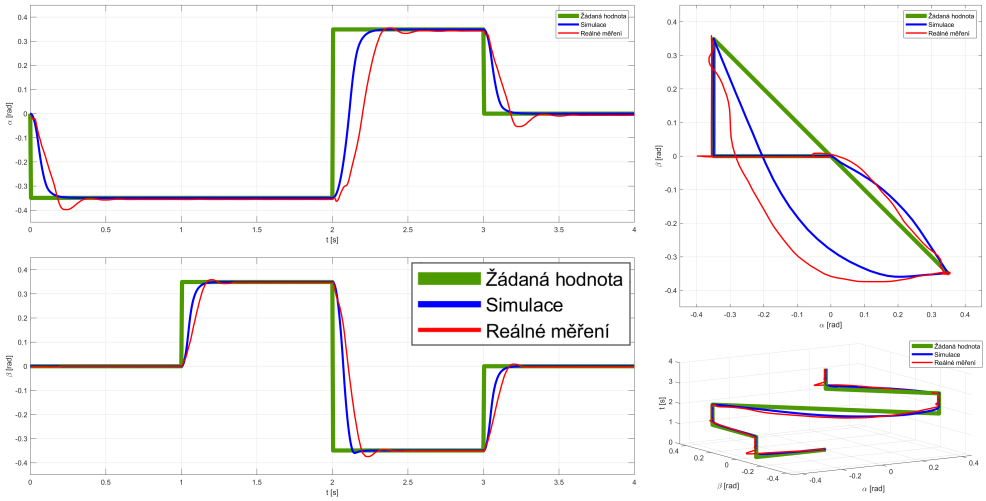
Na Obr. 4.109 jsou zobrazeny průběhy pro RK s periodou celé prováděné sekvence rovnou 4 s. Rozdíly mezi průběhem z fyzikálního a reálného modelu jsou minimální, regulační pochod je plynulý, doba ustálení odpovídá cca 0,5 s a průběhy mají v ustáleném stavu v podstatě nulovou regulační odchylku.



Obr. 4.109 Kaskáda - skoky - T = 4 s

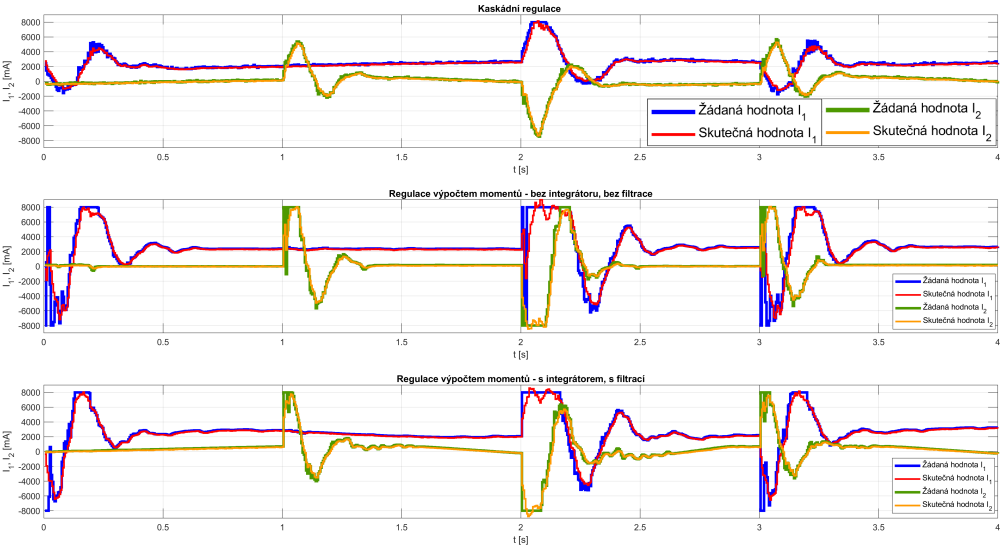
Obr. 4.110 ukazuje chování systému při RVM, a to v původním tvaru bez rozšíření, tedy bez automaticky připínané integrační složky a bez filtrace žádaných hodnot. Již na první pohled je vidět rozdíl mezi fyzikálním a reálným modelem. Ten je způsobena zejména pružností celé sestavy, která není ve fyzikálním modelu přítomna. Při vysoké dynamice a agresivním zásahu regulátoru se již pružnost výrazně projeví, protože není v matematickém modelu zahrnuta. Také je již viditelná trvalá regulační odchylka, hlavně u 1. DOF, zejména kvůli Coulombovu (suchému) tření, které obsahuje každý reálný systém a opět není zahrnuté v matematickém ani fyzikálním modelu.

Při porovnání průběhů fyzikálních modelů obou regulátorů lze pozorovat u RVM více než 2x rychlejší ustálení na žádané hodnotě, protože regulátor využívá proudové limity na maximum - motor vždy naplno zabírá a brzdí. Pružnost reálného systému však zanáší do přechodového děje setrvačnosti, které se regulátoru jeví jako poruchy, a ty je nutné kompenzovat jeho zpětnovazební částí. Tím vznikají



Obr. 4.110 Výpočet momentů - skoky - T = 4 s - bez integrátoru - bez filtrace

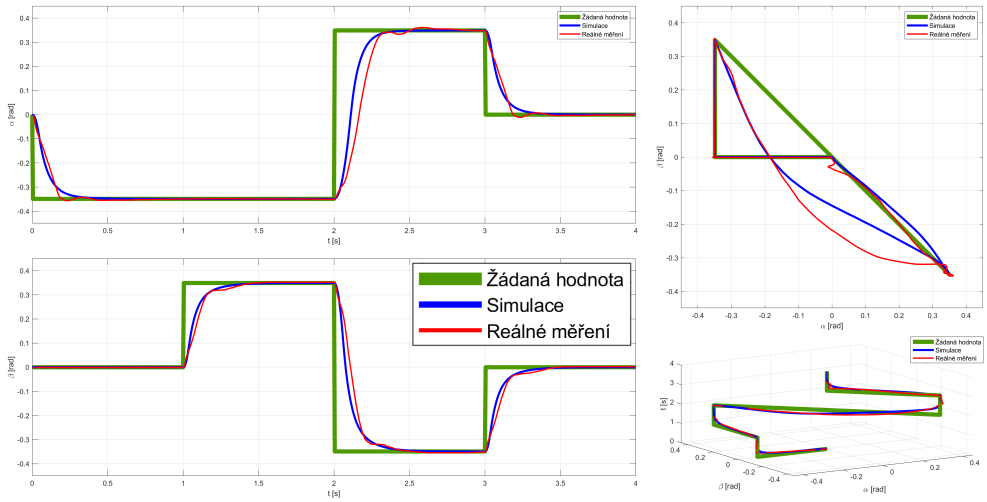
proudové rázy, které přecházejí do proudové rezonance - viz záznam průběhů reálných proudů na Obr. 4.111.



Obr. 4.111 Porovnání průběhů proudů - skoky - T = 4 s

Výše zmíněné neduhy je možné částečně kompenzovat filtrací žádaných hodnot (omezení skokové změny na plynulejší přechod a zamezení tak proudové rezo-

nanci) a připnutím integrační složky ve vhodném okamžiku (minimalizace trvalé regulační odchylky v ustáleném stavu). Výsledek pro toto rozšíření s filtrací a integrátorem je zobrazen na Obr. 4.112 pro úhel náklonu, a na Obr. 4.111 pro odpovídající proudovou odezvu. V tomto případě není změna příliš patrná, ale jisté vyhlazení a odstranění trvalé regulační odchylky pozorovatelné je.

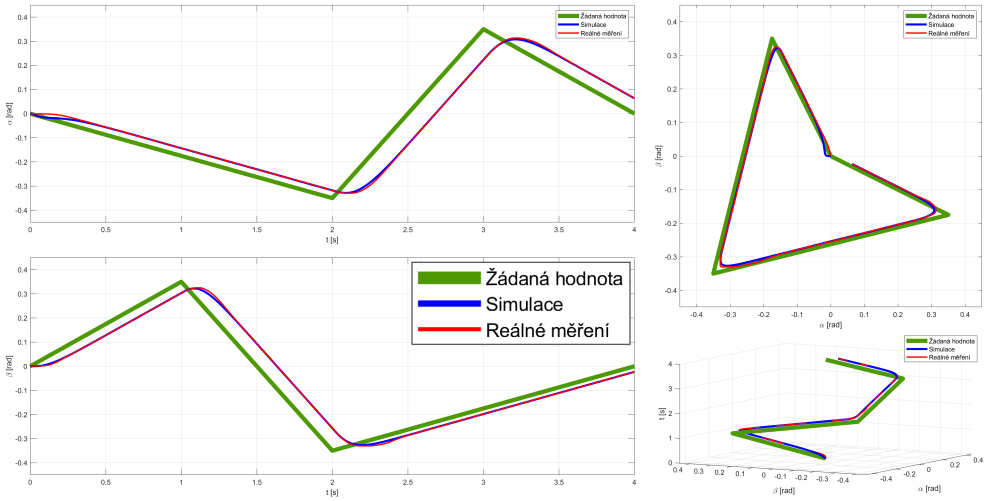


Obr. 4.112 Výpočet momentů - skoky - T = 4 s - s integrátorem - s filtrací

Z testů na reálném systému bylo vypozorováno, že při rychlých skokových změnách žádané hodnoty (např. z nadřazené smyčky pro regulaci polohy kuličky) je nutné filtr žádaných hodnot použít. V opačném případě se dostane systém okamžitě do proudové rezonance. Vliv integrační složky je viditelný zejména při trackingu, což je případ následujících testovacích trajektorií.

Lineární změna žádané hodnoty

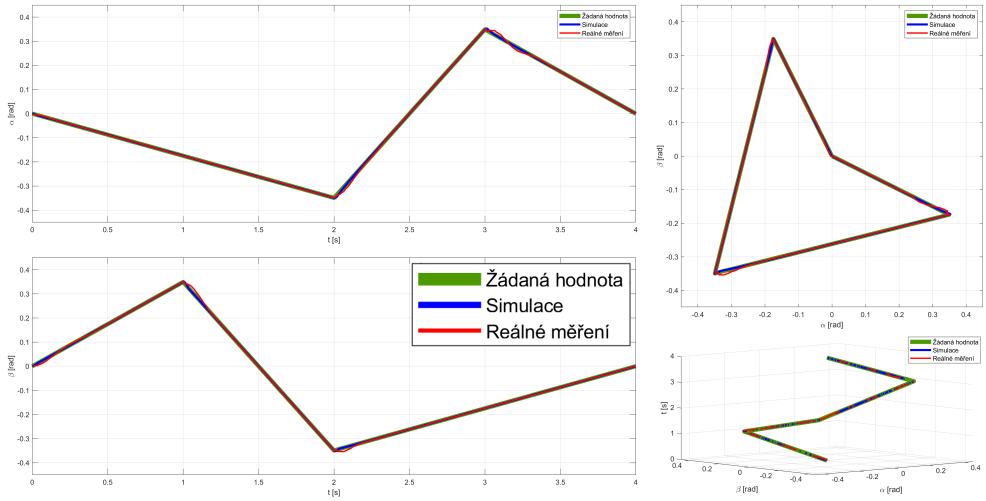
Další logickou testovací křivkou je lineární změna, která je hojně používaná jak při rozjezdových/dojezdových rampách u servomechanismů, tak při požadavku na lineární pohyb aktuátorů. Zde bude kladen důraz na dodržení sklonu přímky (požadavek konstantní rychlosti a tedy nulového zrychlení při ustálení přechodového děje) a především na sledování žádané hodnoty (požadavek trackingu). V tomto případě (a i u dalších typů trajektorií) už lze jednoznačně tvrdit, že RVM dosahuje mnohem lepších výsledků, než RK.



Obr. 4.113 Kaskáda - lineární - T = 4 s

Na Obr. 4.113 pro lineární průběh žádané hodnoty RK je patrný fázový posuv mezi žádanou a skutečnou hodnotou. Na vině není filtr žádané rychlosti, který je v regulační struktuře použitý, ale absence integrační složky v polohové smyčce. Zavedení integrátoru bylo uvažováno i testováno, ale ani při automatickém připínání až v okolí blízkém žádané hodnotě nebylo dosaženo dobrých výsledků. Takovýto systém je sice schopný po ustálení sledovat žádanou hodnotu, ale při každé změně sklonu dojde k velkému překmitu až nestabilitě. U skokové změny žádané hodnoty není tato konfigurace už vůbec použitelná, dochází k překmitům až do dorazu a časová konstanta integrace i hysterezní okno by se muselo pro každý typ i dynamiku trajektorie přenastavovat. Při konstantním nastavení regulátoru pro všechny testovací trajektorie, by zavedení integrační složky výrazně snížilo použitelnost RK a nebude tedy dále uvažováno. Sklon přímky je však dodržen, takže tento požadavek (např. na konstantní rychlost) je splněný, požadavek na tracking už nikoliv.

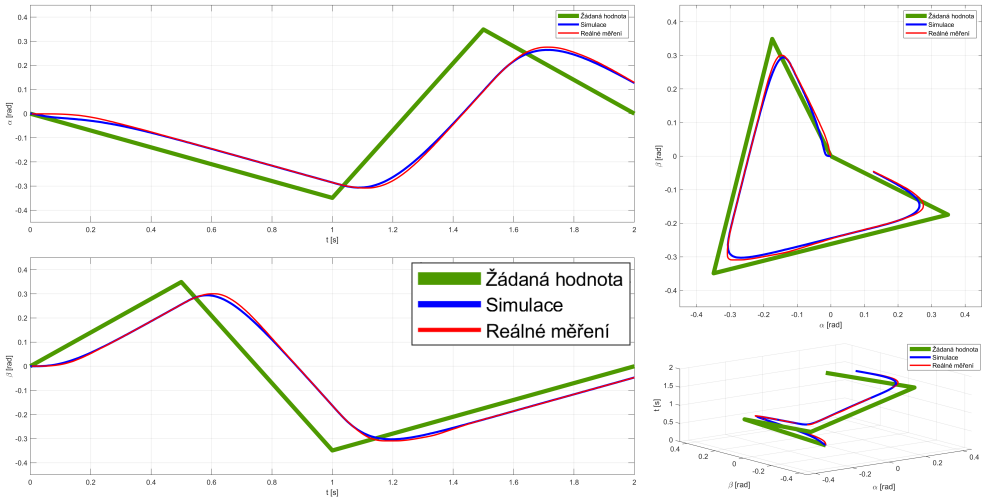
Na Obr. 4.114 je zobrazena odezva RVM na požadavek lineární změny žádané hodnoty. Zde, až na mírný zákmit při změně sklonu přímky, kopíruje skutečný průběh polohy žádanou hodnotu dokonale. Zákmit je pozorovatelný zejména na průběhu z reálného systému (vliv pružnost a suché tření soustavy + integrátor), na průběhu z fyzikálního modelu je zákmit způsoben již pouze vlivem integrační



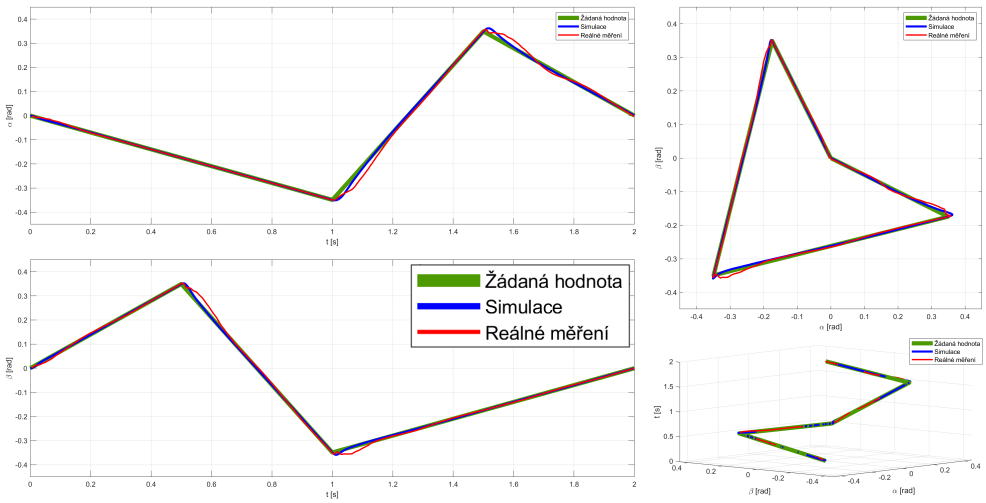
Obr. 4.114 Výpočet momentů - lineární - T = 4 s - s integrátorem - bez filtrace

složky. Tento průběh je dosažený konfigurací regulátoru bez filtrace žádané hodnoty (ta je při trackingu nevhodná, protože posunuje fázi) a s automaticky připínatelnou integrační složkou (minimalizace vlečné chyby). Zde uvedené nastavení je finální a při závěrečném porovnání bude aplikované na všechny testovací trajektorie. Jak by vypadal průběh s integrátorem a s filtrací, nebo bez integrátoru a bez filtrace, je uvedeno v PŘÍLOZE N.

Při zvýšení dynamiky (sklonu) trajektorie (2x rychlejší průběh oproti původnímu), je už integrační složka u RK v podstatě nepoužitelná, protože je vlečná chyba natolik velká, že by musel být integrátor připnutý natrvalo. To způsobí známý efekt integrační složky v polohové smyčce - rozkmitání až nestabilita přechodového děje, dlouhé časy ustálení a pomalá reakce na poruchy. Pro standardní konfiguraci PI regulátor v rychlostní a P regulátor v polohové smyčce je výsledek regulačního pochodu RK uveden na Obr. 4.115.
Průběhy RVM se při zvýšení dynamiky nijak razantně nemění, tracking je však stále splněn, prodlužuje se čas ustálení a zvyšuje překmit. Důvody jsou opět stejné, navíc se začíná objevovat vliv nedostatku točivého momentu, který nejsou schopny aktuátory dodat, aby splnily požadavky na dynamiku žádaného průběhu. Odpovídající grafy jsou zobrazeny na Obr. 4.116.
Další měření se 4-násobným zvětšením sklonu trajektorie oproti původní, jsou



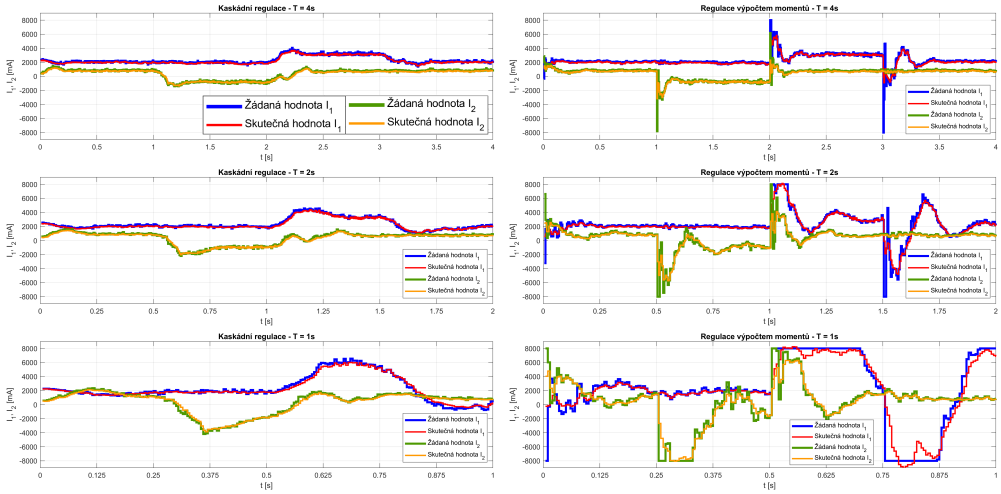
Obr. 4.115 Kaskáda - lineární - T = 2 s



Obr. 4.116 Výpočet momentů - lineární - T = 2 s

uvedeny v PŘÍLOZE O. Na přiložených grafech pro RVM je stále vidět dobré sledování žádané hodnoty, s přihlédnutím na momentové limity celého systému. Pružnost sestavy se zde již viditelně projevuje, zejména v 1. DOF. U průběhů z RK se již nedá mluvit ani o dodržení sklonu přímky, protože rychlost přechodového děje je natolik vysoká, že nedojde ani k jeho ustálení. U RK je však i u

takto vysoké dynamiky stále proudová/momentová rezerva, přičemž RVM se dostává na proudové/momentové limity i u nejpomalejšího průběhu. Důkaz těchto tvrzení je demonstrován v grafech proudových odezev pro různou požadovanou dynamiku na Obr. 4.117.

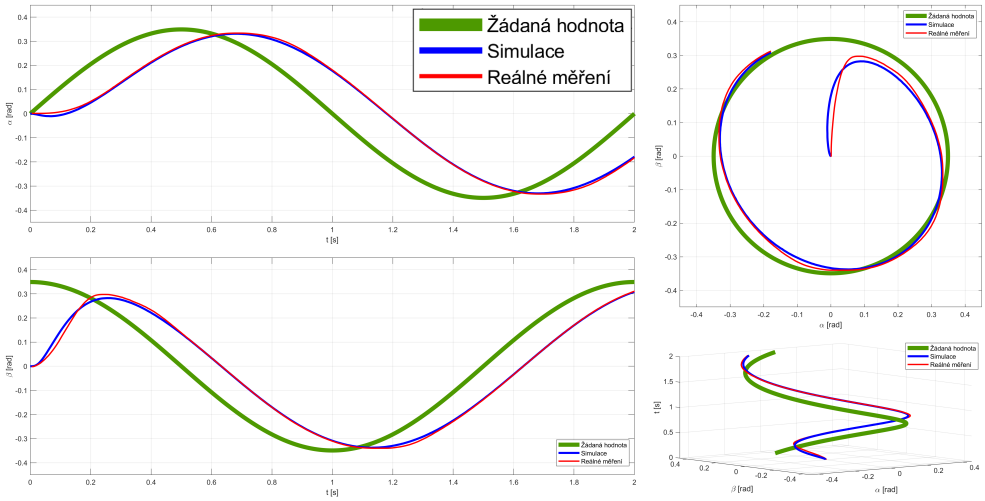


Obr. 4.117 Porovnání průběhů proudů při lineární trajektorii

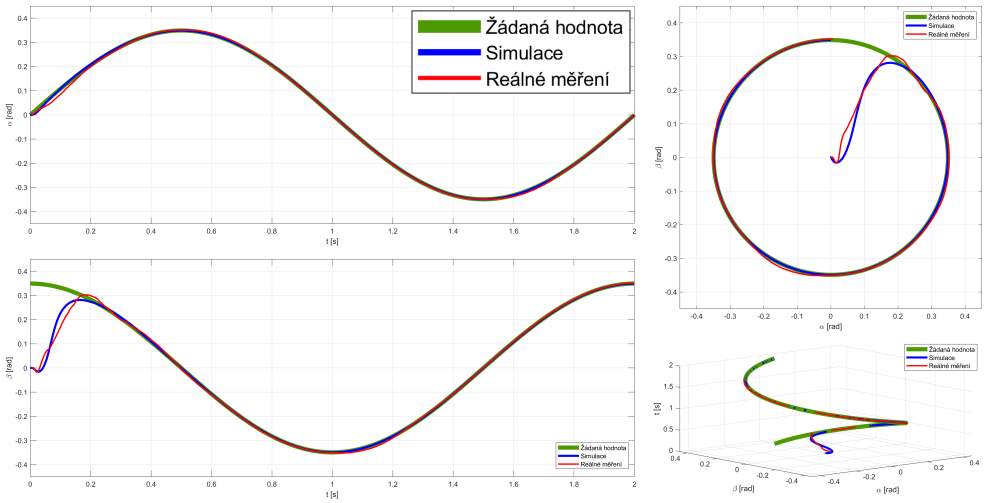
Harmonická změna žádané hodnoty - kružnice

Předposlední testovací trajektorií je harmonická změna obou úhlů náklonu tak, aby při závislosti jednoho úhlu na druhém tvořila kružnici. Oproti lineární trajektorii jsou zde všechny stavové proměnné nenulové v celém rozsahu měření a mají harmonický charakter. Opět v tomto případě je zjevné, že RVM dosahuje mnohem lepších výsledků, než RK, a to i ve sledování žádané trajektorie s minimální vlečnou chybou při vysoké dynamice pohybu.
Výsledky pro RK jsou zobrazeny na Obr. 4.118 a odpovídají střední dynamice průběhu s jednou periodou harmonické funkce rovnou 2 s. Fázový posuv byl již očekávaný, po ustálení přechodového děje odpovídá cca 35° pro 1. DOF a 30° pro 2. DOF, amplituda výstupního signálu poklesla zhruba o 5% a tvar kružnice se začíná deformovat do elipsy.

Na Obr. 4.119 jsou průběhy z RVM, zde je dodržena amplituda i fáze žádaného průběhu s minimální trvalou regulační odchylkou. Vliv pružnosti je viditelný jen

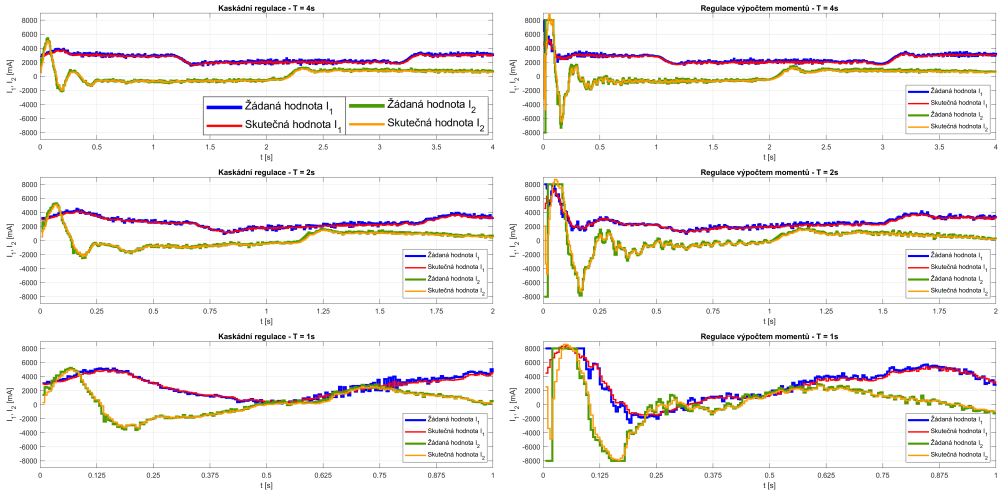


Obr. 4.118 Kaskáda - kružnice - T = 2 s



Obr. 4.119 Výpočet momentů - kružnice - T = 2 s

v počátečním přechodovém ději se skokovou změnou žádané hodnoty u 2. DOF, po ustálení už dochází k plynulým přechodům vstupního signálu, a proto je příspěvek k poruše od flexibility materiálu v podstatě zanedbatelný. Ostatní měření pro periodu 4 s a 1 s jsou uvedeny v PŘÍLOZE P. Proudová odezva všech probíraných průběhů je vykreslena na Obr. 4.120.

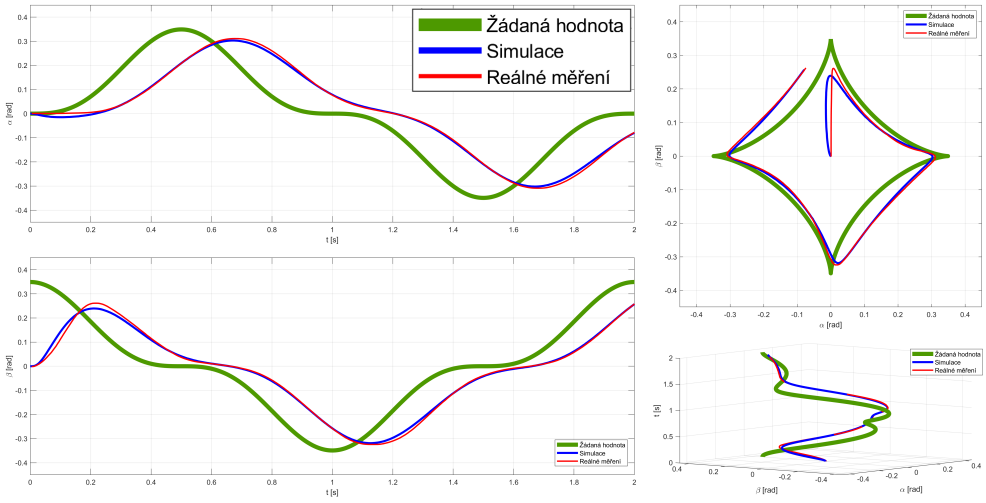


Obr. 4.120 Porovnání průběhů proudů při trajektorii kružnice

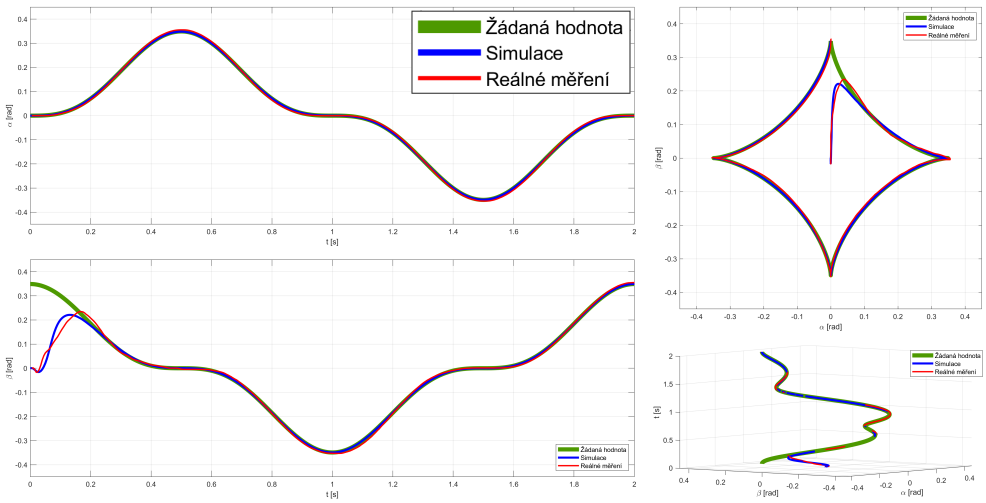
Jak vidno, po odeznění počátečního skokového přechodového děje není ani u jednoho z regulátorů dosažen proudový limit a průběhy jsou dosti podobné. V případě kružnice by tedy bylo možné ještě přidat na dynamice bez zjevného vlivu na kvalitu regulace (platí především pro RVM).

Neharmonická změna žádané hodnoty - asteroida

Poslední testovací trajektorie je nazvaná jako asteroida, což je neharmonická změna obou úhlů náklonu tak, aby při závislosti jednoho na druhém tvořila hypocykloidu se 4 vrcholy. Je to tedy složenina dvou harmonických signálu pro každý úhel, kdy se ve výsledku všechny stavové proměnné neharmonicky mění a reprezentuje průběh s největší dynamikou pohybu. I v tomto případě lze soudit, že RVM dosahuje jednoznačně lepších výsledků, než RK. Platí totéž co u kružnice - sledování žádané trajektorie je splněno s minimální vlečnou chybou a po ustálení dosahuje regulace i při vysoké dynamice vynikajících výsledků, v porovnání s RK. Vše za předpokladu, že je k dispozici dostatečný točivý moment.
Na Obr. 4.121 je zobrazen průběh se střední dynamikou pohybu pro RK. Fázové posuvy i snížení amplitud jsou zhruba stejné jako u kružnice a tvar křivky se začíná deformovat do obdélníku.
Grafy na Obr. 4.122 pro RVM dodržují amplitudu i fázi a dokazují schopnost



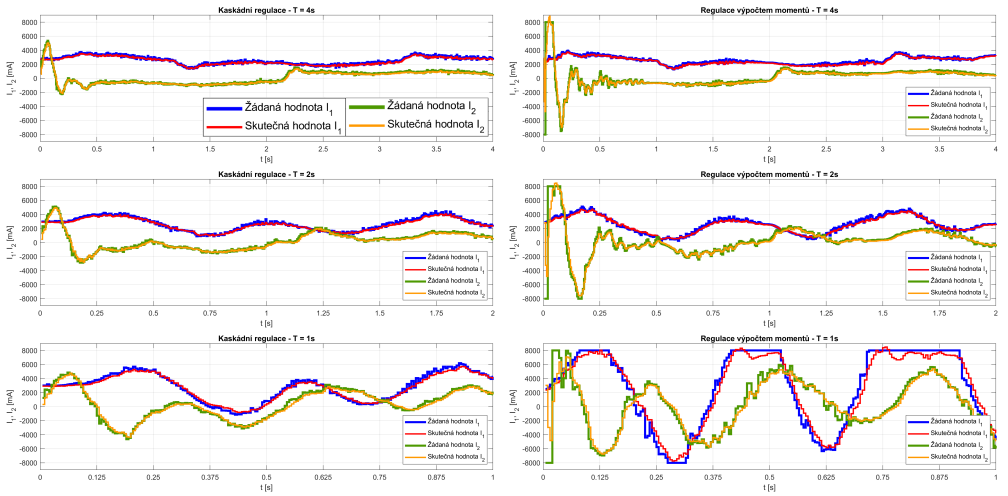
Obr. 4.121 Kaskáda - asteroida - T = 2 s



Obr. 4.122 Výpočet momentů - asteroida - T = 2 s

regulátoru sledovat žádanou hodnotu s minimální vlečnou chybou. Platí zde v podstatě vše co pro kružnici, ostatní měření pro periodu 4 s a 1 s jsou uvedeny v PŘÍLOZE Q. Zajímavý je zejména průběh Q.c, kde už v druhé části grafu vypadá, že regulátor nezvládá zajistit požadovaný průběh. Příčinou je nedostatečný proud/moment aktuátorů, který není řídicí jednotka schopna dodat. Tento

proud (8 A) je omezen také v GUI aplikaci jako maximální možný žádaný proud vstupující do servozesilovače, 8 A je jmenovitý proud obou aktuátorů. Řídicí jednotka je však schopna dodat krátkodobě (po dobu cca 100 ms) až 4-násobek nominálního proudu. V PŘÍLOZE Q.e) je vykreslený i průběh, kdy je povolen 2-násobný (16 A) žádaný proud, než ve standardním případě. Jak vidno, sledování žádané hodnoty je opět zajištěno, zde už ale ke zvýšené vlečné chybě viditelně přispívá pružnost. Na grafu Q.f následuje i ověření, že zjednodušený i přesný model dosahují v podstatě totožných výsledků a pro maximální dosažitelnou dynamiku tohoto systému tedy plně postačuje zjednodušený matematický popis. Navýšení povoleného proudu pro RK nemá smysl, protože zde nebylo proudových limitů vůbec dosaženo - viz Obr. 4.123, spolu s ostatními proudovými charakteristikami.

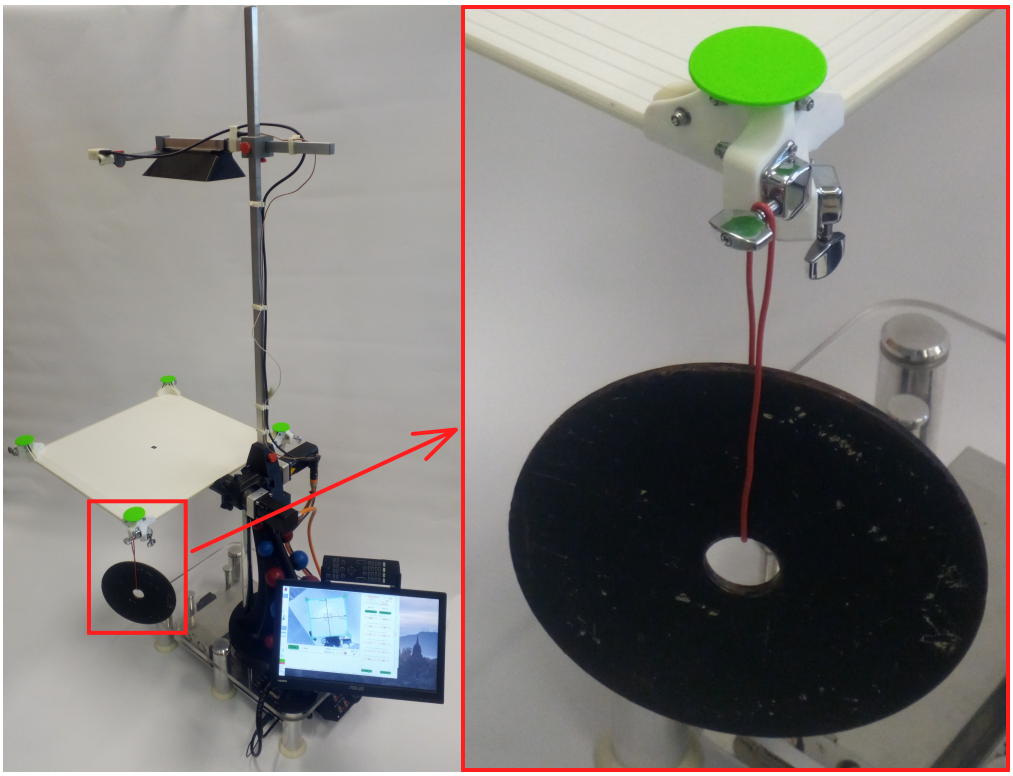


Obr. 4.123 Porovnání průběhů proudů při trajektorii asteroida

Měření při trvalé poruše - závaží

Všechny testovací trajektorie byly znovu přeměřeny stejným způsobem, se zavěšeným závažím o hmotnosti 1 kg, které demonstrovalo trvalou poruchu systému, resp. významnou chybu v matematickém modelu. Upevnění nebylo v reálných podmínkách pevné, proto navíc generuje do měřených dat chybu od volného zavěšení - oscilace závaží. Náhled na konkrétní upevnění je na Obr. 4.124, ve

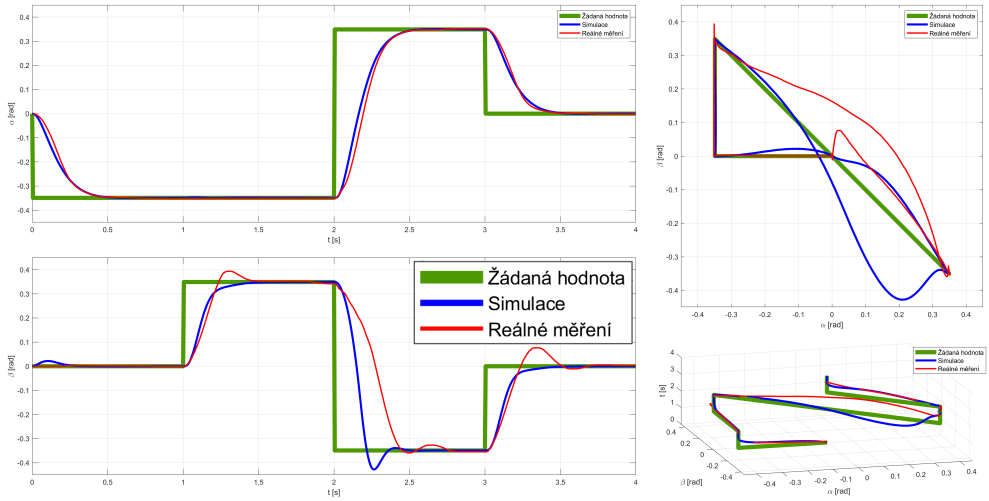
fyzikálním modelu byla do místa uchycení přidána odpovídající hmota.



Obr. 4.124 Místo zavěšení závaží při měření na reálném modelu

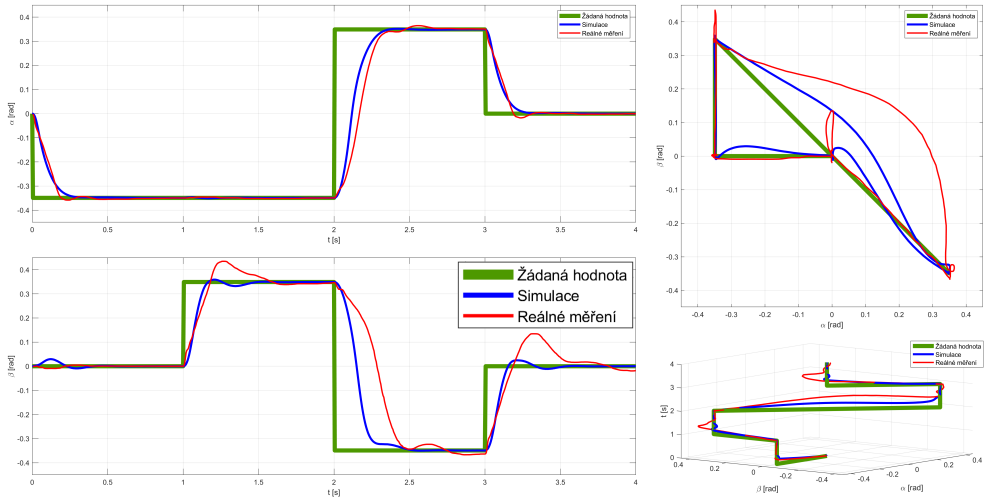
Vzhledem na umístění závaží a kinematické uspořádání modelu je nutno podotknout, že na první DOF by neměla mít tato změna výrazný vliv, avšak na druhý DOF přesně naopak. Struktura je totiž z pohledu 1. DOF nevyvážený i bez přidaného závaží, které zvýší asymetrické rozložení hmot o cca 25%. Uspořádání pro 2. DOF však bylo před přidáním závaží vyvážené a nyní je potřeba zhruba 50% dostupného točivého momentu pouze na stabilizaci ve výchozí pozici. Jinými slovy, matematický model pro 1. DOF se změnil pouze mírně, pro 2. DOF ale zásadně. Níže jsou uvedeny pouze vybrané průběhy, kompletní záznam je doplněn v PŘÍLOZE R.

První testy proběhly opět na skokové změně žádané hodnoty. Pro RK lze pozorovat znatelné změny oproti situaci bez závaží pouze u 2. DOF, a to v překmitech a prodloužené době ustálení - viz Obr. 4.125.



Obr. 4.125 Kaskáda - skoky - závaží - T = 4 s

Podobných výsledků bylo dosaženo i u RVM - 1. DOF vykazuje pouze mírnou odchylku oproti situaci bez závaží, 2. DOF má naopak velký překmit a dlouhou dobu ustálení (pokud se dá vůbec o ustálení mluvit), kterého je dosaženo pouze díky připínané integrační složce ve 3° okně okolo žádané hodnoty (Obr. 4.126).

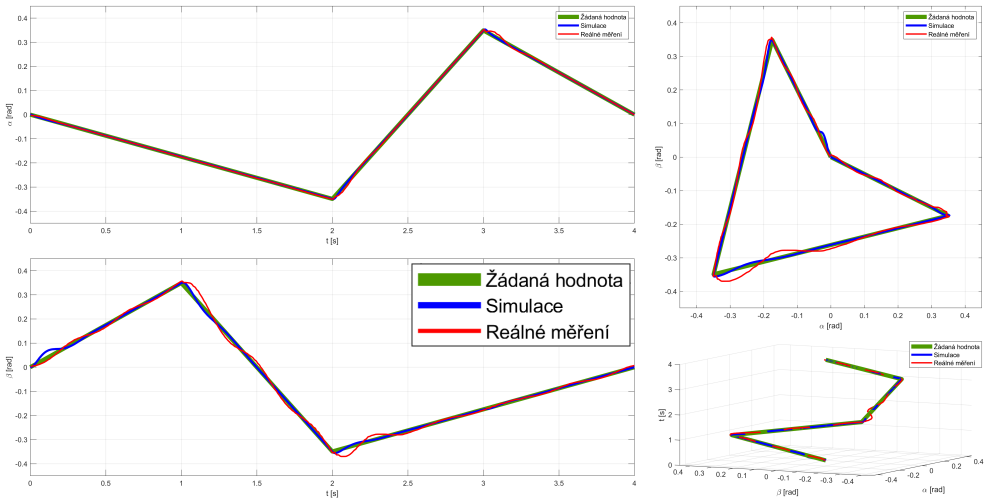


Obr. 4.126 Výpočet momentů - skoky - závaží - T = 4 s - $Hyst_{Int}$ = 3°

Porovnání se zmenšeným aktivním pásmem integrátoru je uvedeno v PŘÍLOZE

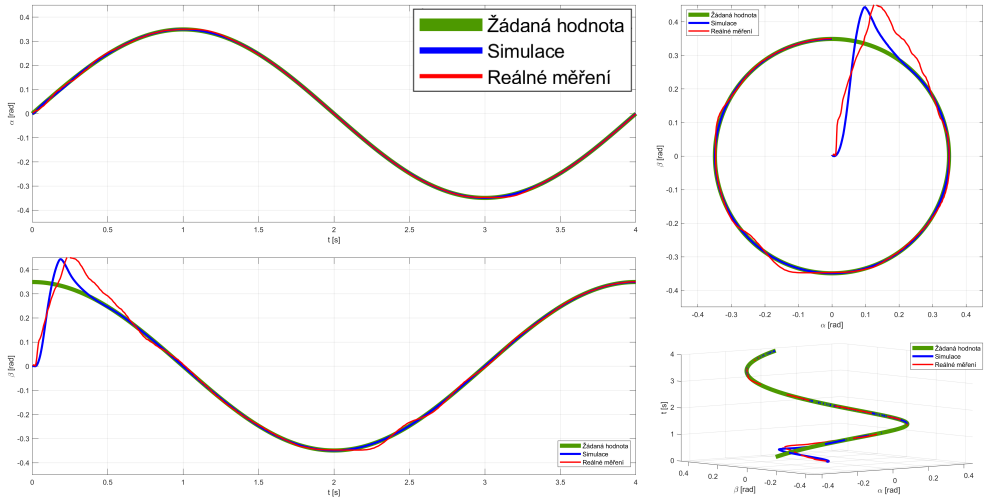
R.a, kde už zůstává trvalá regulační odchylka. Při vzájemném srovnání dosahuje RK při skokové změně žádané hodnoty opět lepších výsledků, než RVM. U dalších testovacích trajektorií je tomu přesně naopak, ovšem za předpokladu, že je zařazena automaticky připínatelná integrační složka a vhodné aktivní pásmo (hystereze). Při jejím vyřazení by vznikla trvalá regulační odchylka, způsobená samotnou podstatou fungování RVM, který vychází z přesné znalosti dynamických rovnic popisujících chování systému.

Vybraný průběh pro lineární změnu žádané hodnoty je zobrazen na Obr. 4.127 a doplňující grafy jsou v PŘÍLOZE R.b,c,d. U 2. DOF pro RVM je kmitání způsobené zejména integrační složkou a volným uložením závaží, u RK jsou průběhy, kromě drobných zákmitů u reálného měření v 2. DOF, v podstatě totožné s měřením bez závaží. Testování neprobíhalo pro maximální dynamiku pohybu T = 1 s, protože při prvních pokusech došlo vlivem momentových rázů a zvýšené setrvačnosti roviny ke zlomení nosného ramene v místě uchycení - PŘÍLOHA R.k

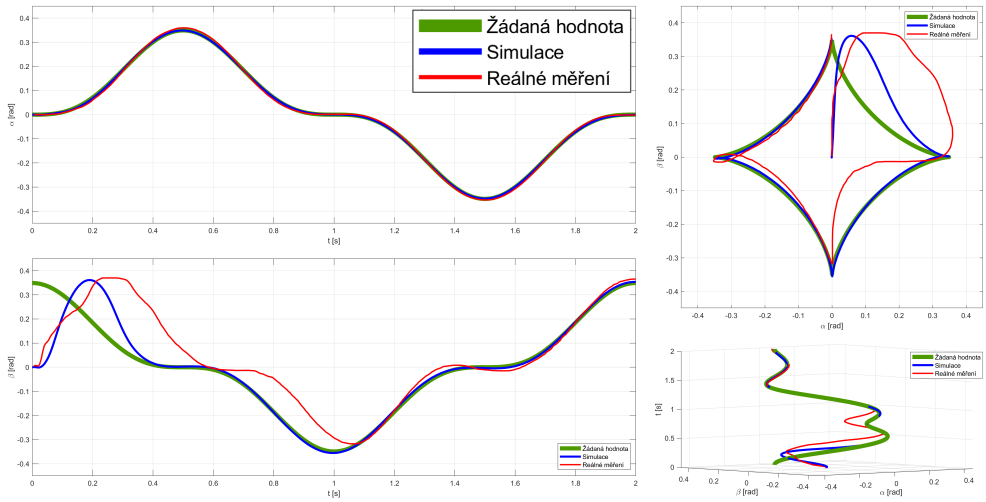


Obr. 4.127 Výpočet momentů - lineární - závaží - T = 4 s

U harmonického testovacího průběhu uvedeného na Obr. 4.128 a v PŘÍLOHÁCH R.e,f,g jde vidět pro RVM velmi dobré sledování žádané hodnoty se zákmitem v 2. DOF při změně směru rychlosti, u RK jsou průběhy až na rozkmitání v 2. DOF od volného uložení totožné, jako pro situaci bez závaží - skutečná hodnota fázově posunutá oproti žádané.



Obr. 4.128 Výpočet momentů - kružnice - závaží - T = 4 s



Obr. 4.129 Výpočet momentů - asteroida - závaží - T = 2 s

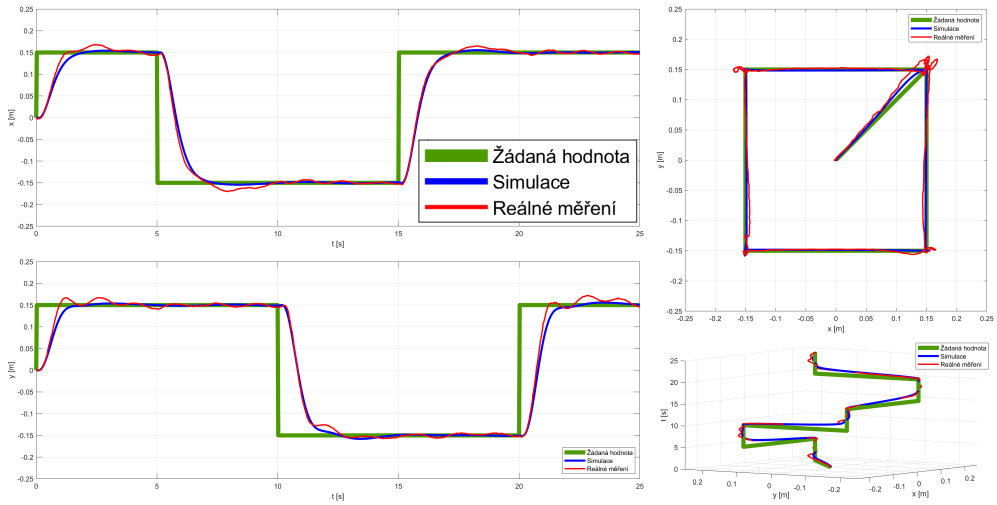
Poslední testovaný neharmonický průběh sleduje žádanou hodnotu obstojně, pokud se pohybuje v aktivním pásmu integrační složky. V opačném případě má trvalou regulační odchylku, která se začne minimalizovat až po průchodu hysterezním oknem - Obr. 4.129. Doplňující grafy jsou zobrazeny v PŘÍLOZE R.h,i,j.

4.8.8 Porovnání regulátorů při regulaci kuličky

Při použití RK a RVM na řízení polohy kuličky na nakloněné rovině nejde ani tak o porovnání regulátorů samotných, jako o porovnání, jak tyto regulátory fungují s využitím výstupní hodnoty nadřazeného PD(I) regulátoru kuličky. Tato regulační struktura byla již odladěná a otestována v podkapitole 4.8.4 s využitím RK. Pro vyhodnocení a porovnání s RVM byla do testovacích trajektorií přidána lineární změna žádané hodnoty a celé měření proběhlo pro dříve nastavenou PD(I) regulaci kuličky s neměnným nastavením RK i RVM. Pro každou testovací trajektorii bylo zvoleno jedno nastavení, s kterým bylo na reálném modelu zaznamenáno 10 opakovaných měření. Vykreslovaný průběh byl potom výsledkem statistického vyhodnocení získaných dat, a spolu s průběhem z měření na fyzikálním modelu byl následně porovnáván s ostatními charakteristikami.

Při frekvenci zpracování snímků z kamery cca 40 Hz (FPS), charakteru žádané hodnoty (skoková změna), použití nadřazeného PD(I) regulátoru kuličky a všech ostatních již dříve zmiňovaných limitů reálného systému lze konstatovat, že je u RVM nutné povolit filtr žádaných hodnot. V opačném případě dojde ve všech případech k proudové rezonanci, protože RVM nemá žádnou setrvačnost a reaguje na skokové změny žádané hodnoty okamžitým požadavkem na proud, ve většině případů s hodnotou odpovídající proudovému limitu. Než ale stačí přechodový děj odeznít, přijde od nadřazeného regulátoru další požadavek generující RVM zásah v blízkosti proudového limitu s opačným znaménkem, než tomu bylo v předchozím kroku. Tím se systém dostane do stavu blížícímu se průběhům na Obr. 4.73 a 4.74. Systém je stále stabilní, popř. se blíží k hranici stability, avšak mechanika zařízení je velice namáhána a dlouhodobě toto zacházení nesnese. Použitý filtr žádaných hodnot zabrání těmto skokovým změnám, náběh proudu je potom pozvolný, a fázový posuv signálu způsobený tímto filtrem je, s přihlédnutím na dynamiku regulace kuličky, v podstatě zanedbatelný.

Průběhy se skokovou a lineární změnou žádané hodnoty se při porovnání nejvíce lišily, i tak zde ale byl rozdíl minimální. Skoková změna ve tvaru čtverce (při promítnutí do x-y závislosti) je pro RK zobrazena na Obr. 4.130, pro RVM na Obr. 4.131. Jak vidno, jedná se především o plynulejší průběh bez větších

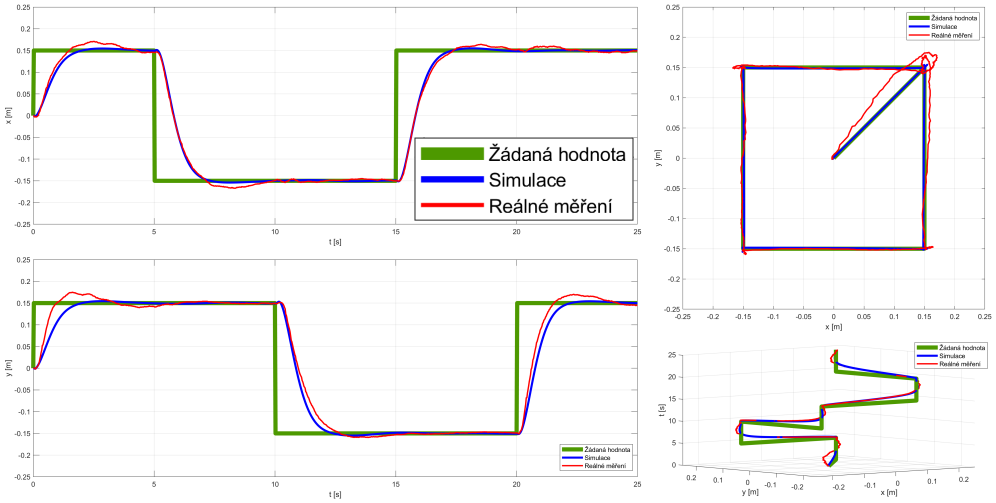


Obr. 4.130 Regulace kuličky s RK - čtverec - T = 25 s

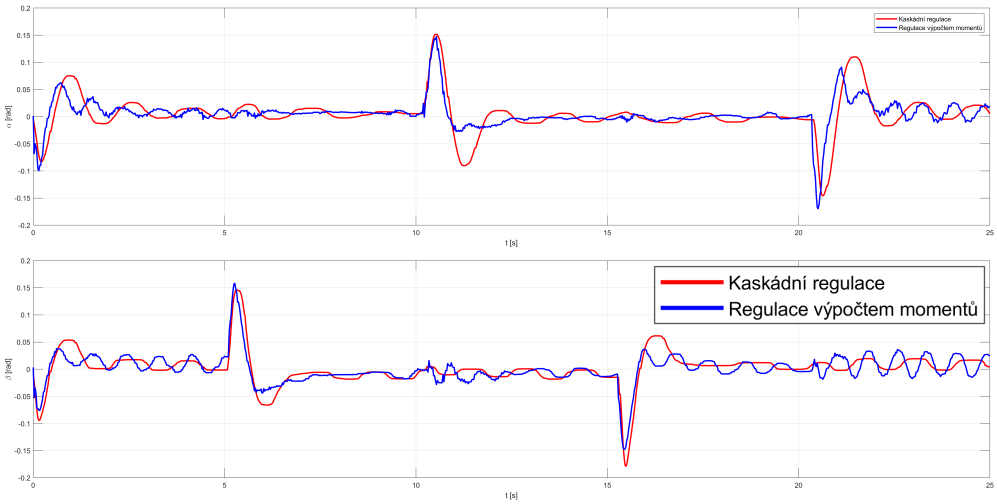
zákmitů ve prospěch RVM, což je vzhledem k chování náklonu roviny při regulaci zarážející. Vysokofrekvenční kmitání proudu charakteristické pro metodu RVM, se v použité mechanické struktuře projeví jako rezonance náklonu roviny v rámci vůlí v uložení, která je snímači polohy detekovatelná jen omezeně. Z měření úhlů náklonů pro trajektorii čtverce je zmiňovaná rezonance patrná, viz Obr. 4.132.

S ohledem na charakter propojení řízeného objektu (kuličky) s nakloněnou rovinou (pouze přes rozklad vektoru gravitace), je pro skokovou změnu žádané hodnoty vhodnější RK, i když lepších výsledků bylo dosaženo s využitím RVM. Agresivní projevy RVM při skokové změně zapříčiňují odskoky kuličky v ose z, které jsou neměřitelné a patrné pouze z reálného pozorování. Další negativní vlastností RVM je z pohledu mechanického uspořádání rezonance úhlů náklonů, která způsobuje s vůlemi v uložení mechanické rázy ve styčných bodech převodového soukolí a rapidně tak snižuje životnost mechanických částí systému.

Grafy na Obr. 4.133 a 4.134 ukazují chování obou regulátorů při lineární změně žádané hodnoty. U RK průběh více kmitá, ale frekvence kmitů je menší, než u RVM, kde je křivka z pohledu polohy kuličky více vyhlazená a podobnost s průběhem z fyzikálního modelu je větší. Rozdíly mezi oběma metodami už však začínají mizet, přesto je u průběhu s postupnou změnou žádané hodnoty

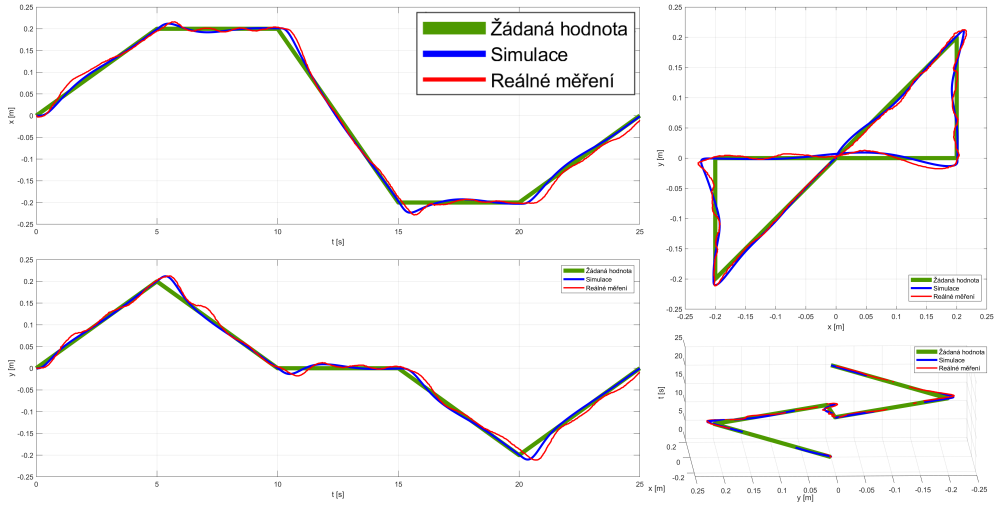


Obr. 4.131 Regulace kuličky s RVM - čtverec - T = 25 s



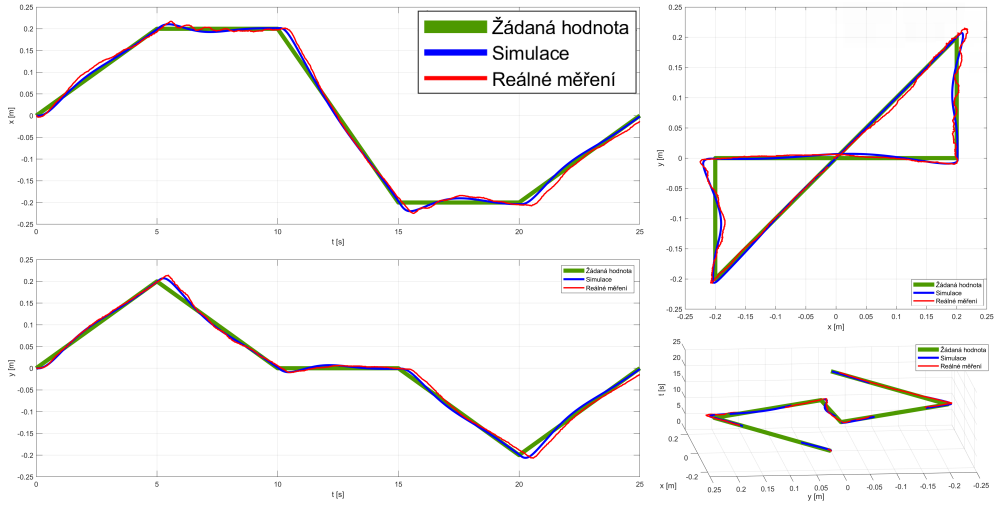
Obr. 4.132 Regulace kuličky s RK a RVM - čtverec - T = 25 s - porovnání úhlů náklonu

vhodnější využít RVM. Oproti skokové změně už nedochází k prudkým proudovým/mechanickým rázům, kmitání s vyšší frekvencí okolo žádané hodnoty (oproti RK) stále zůstává přítomno, ale regulátor není tak utlumený jako RK a snaží se splnit požadavky z nadřazeného regulátoru rychle a za každou cenu.



Obr. 4.133 Regulace kuličky s RK - lineární - T = 25 s

Z pohledu sledování žádaného náklonu roviny je u RK opět viditelný fázový posuv, jak bylo demonstrováno v předchozí kapitole 4.8.7, avšak filtrem žádaných hodnot byl taktéž uměle zaveden fázový posuv i do RVM, i když ne tak velký, jako u RK. Integrační složka v nadřazeném PD(I) regulátoru kuličky však zajistí postupné ustálení na žádané hodnotě u obou metod.



Obr. 4.134 Regulace kuličky s RVM - lineární - T = 25 s

Zbývající průběhy pro kružnici a asteroidu vykazují podobný charakter jako pro lineární změnu, rozdíly mezi RK a RVM jsou však minimální a volba lepší varianty už není jednoznačná a závisí na požadavcích na regulační pochod. Jednotlivé průběhy jsou uvedeny v PŘÍLOZE S.

4.9 Závěrečné vyhodnocení

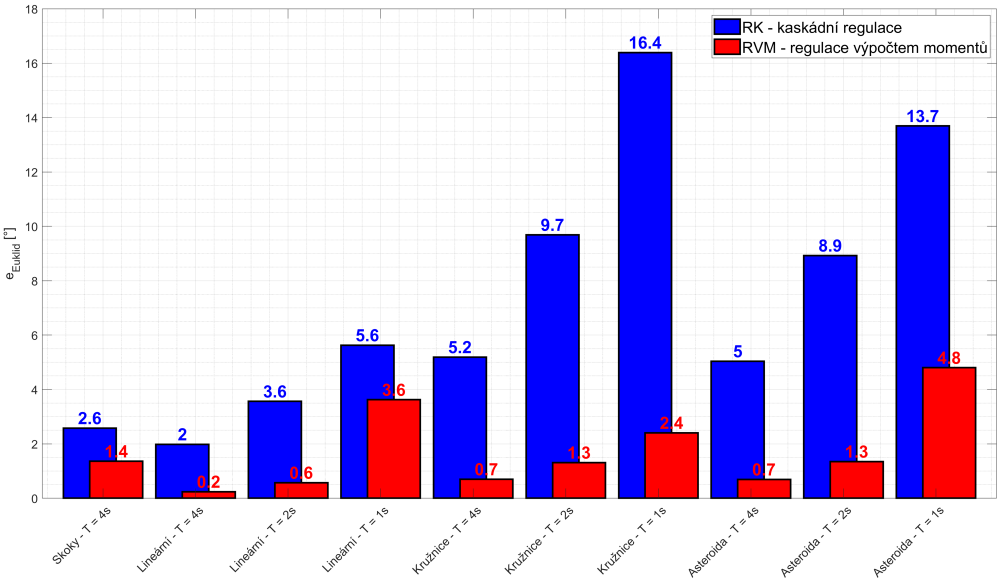
V této podkapitole bude popsáno závěrečné vyhodnocení probraných přístupů k řízení pohybu. Kvalita regulace bude posuzována z více úhlů pohledů, kterými jsou konkrétně:

- Regulační odchylka (průměrná Euklidovská odchylka).
- Rychlost/doba regulace (čas dosažení 95% žádané hodnoty).
- Nároky na elektrickou energii (procentuální porovnání).
- Podobnost chování reálného systému vs fyzikálního modelu při regulaci.
- Mechanické zatížení systému.

Veškeré porovnání je navázáno přímo na RK a RVM, tedy na regulaci náklonu roviny, ne na regulaci pozice kuličky na nakloněné rovině. Na regulační pochod kuličky má majoritní vliv její PD(I) regulátor, takže zde se nedá mluvit o porovnání RK a RVM, ale spíše o porovnání součinnosti těchto regulátorů s PD(I) regulátorem, což není cílem tohoto vyhodnocení a bude uvedeno pouze jako doplněk.

4.9.1 Porovnání z pohledu regulační odchylky

Aby byly výsledky porovnání všech testovacích trajektoriích jasné a názorné, tedy zobrazitelné ideálně v jednom grafu, je z každého průběhu vypočtena průměrná Euklidovská odchylka a postupně vložena do sloupcového grafu. Tento graf je zobrazen na Obr. 4.135 a obsahuje vyhodnocení průběhů probíraných v kapitole 4.8.7.
Při porovnání kvality regulace z pohledu regulační odchylky je patrné, že RVM dosahuje ve všech měřených případech lepších výsledků, než RK. Euklidovská

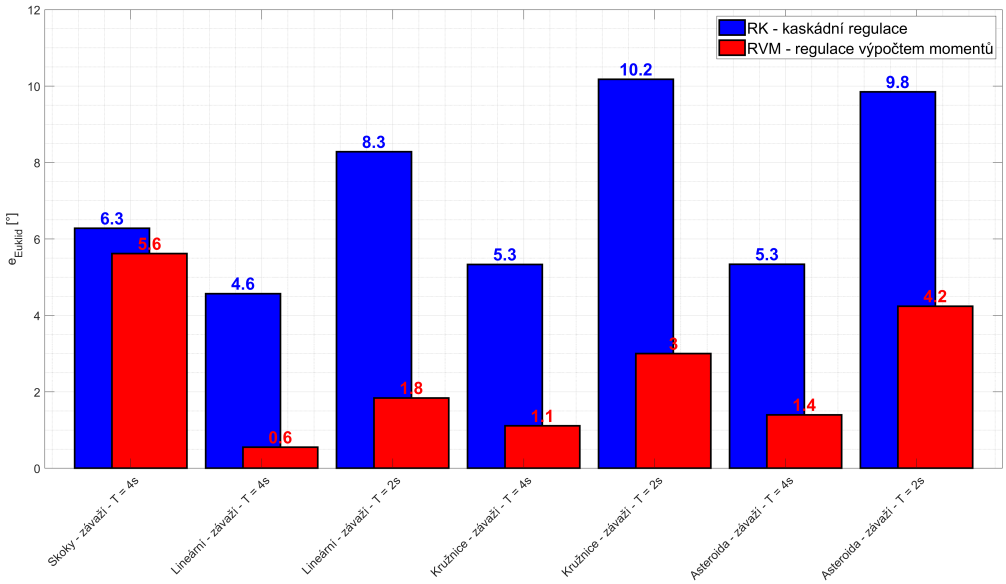


Obr. 4.135 Porovnání RK s RVM na základě Euklidovské regulační odchylky

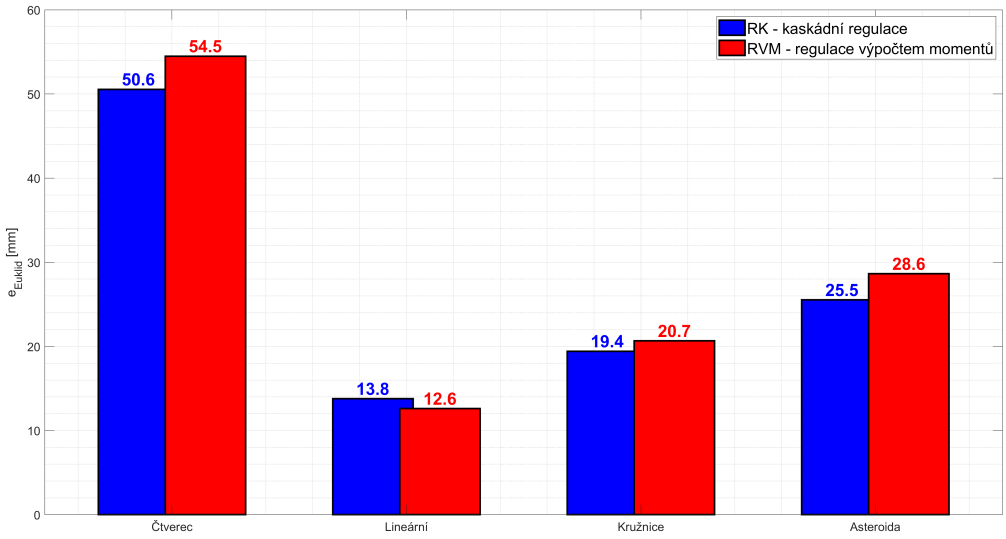
regulační odchylka u RK se pohybovala v rozmezí 1,5 až 10 násobku RVM, s průměrnou hodnotou cca 4,3.

Nejhorších výsledků bylo u RVM dosaženo při skokové změně žádané hodnoty a při lineární změně s maximální dynamikou, kde už nebyl dostupný točivý moment, což je dokázáno na grafu s měřenými proudy - Obr. 4.117. Naopak nejlepší odezvu měl tento regulátor na průbězích s pozvolnou změnou žádané hodnoty s nízkou dynamikou pohybu - lineární, kružnice a asteroida s periodou opakování 4 s.

Podobné závěry platí i pro měření s uměle zavedenou trvalou poruchou systému (se závažím), jen s tím rozdílem, že se zmenšuje odchylka mezi jednotlivými regulátory. Pořád je zde však znatelný nepoměr, v průměru má RK 2,8 krát větší průměrnou regulační odchylku, než RVM - viz Obr. 4.136
RK vykazuje nejhorší chování u průběhů s vysokou dynamikou pohybu, kde je na vině velký fázový posuv a velmi pomalá reakce regulátoru na změnu žádané hodnoty. Nejlepších výsledků bylo dosaženo u skokové změny žádané hodnoty, kde je fázový posuv a pomalá reakce regulátoru naopak výhodou.
Na Obr. 4.137 je také uvedeno porovnání výsledků regulace kuličky s PD(I)



Obr. 4.136 Porovnání RK s RVM Euklidovskou regulační odchylkou - závaží



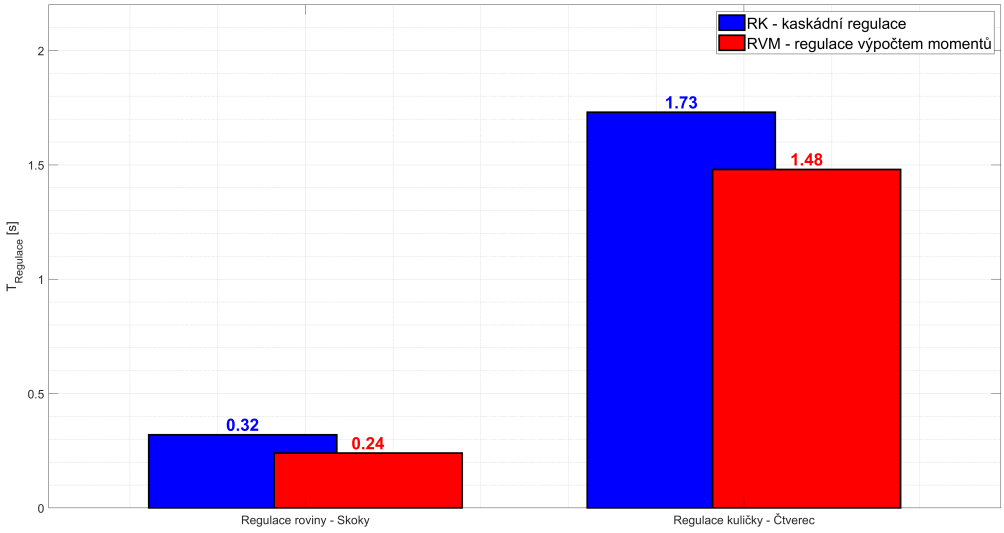
Obr. 4.137 Porovnání RK s RVM Euklidovskou regulační odchylkou při regulaci kuličky

regulátorem, s využitím podřízeného regulátoru typu RK a RVM. Jak bylo již několikrát zmíněno, nejedná se o porovnání RK a RVM jako takových, ale o jejich

použitelnost v součinnosti s nadřazeným PD(I) regulátorem z pohledu Euklidovské regulační odchylky. Výsledky jsou téměř totožné, u trajektorie typu *Čtverec*, *Kružnice* a *Asteroida* dosahuje lepších výsledků RK, u *Lineárního* průběhu zase RVM. S přihlédnutím i na průběhy v kapitolách 4.8.8 a 4.8.7 lze celkově říci, že RK je lepší volbou pro plynulý náklon roviny a celkovou robustnost, RVM zase vykazuje lepší chování při požadavcích na plynulý pohyb kuličky, protože při pohybu tak nekmitá, jako u RK.

4.9.2 Porovnání z pohledu rychlosti/doby regulace

Vzhledem k charakteru testovacích trajektorií a fázovému posuvu RK při trackingu, byla jako referenční trajektorie vybrána skoková změna žádané hodnoty u regulace roviny a *Čtverec* u regulace kuličky. V opačném případě by pro většinu průběhu u RK nebylo 95% žádané hodnoty ani dosaženo (u regulace roviny) a porovnání by bylo značně zkreslené. Z odpovídajících průběhů byla určena průměrná doba regulace z celého rozsahu a výsledné porovnání je uvedeno na Obr. 4.138.



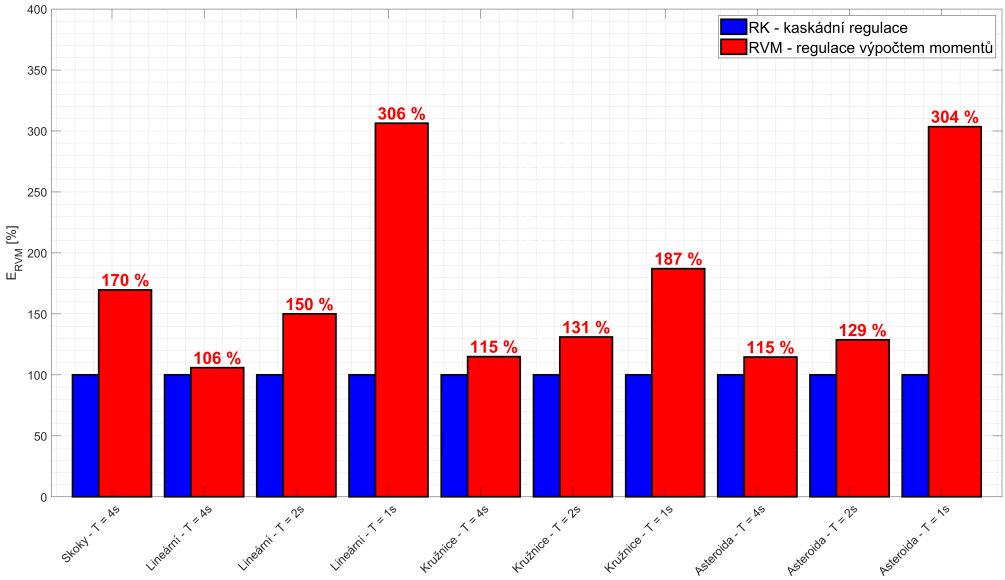
Obr. 4.138 Porovnání RK a RVM z pohledu času regulace

Při regulaci náklonu roviny je vidět, že RVM je zhruba o 1/4 rychlejší, než RK. Při regulaci kuličky už není mezi RK a RVM takový rozdíl, protože hlavní vliv

na čas regulace má nadřazený PD(I) regulátor. RK je však stále o cca 15% pomalejší, než RVM.

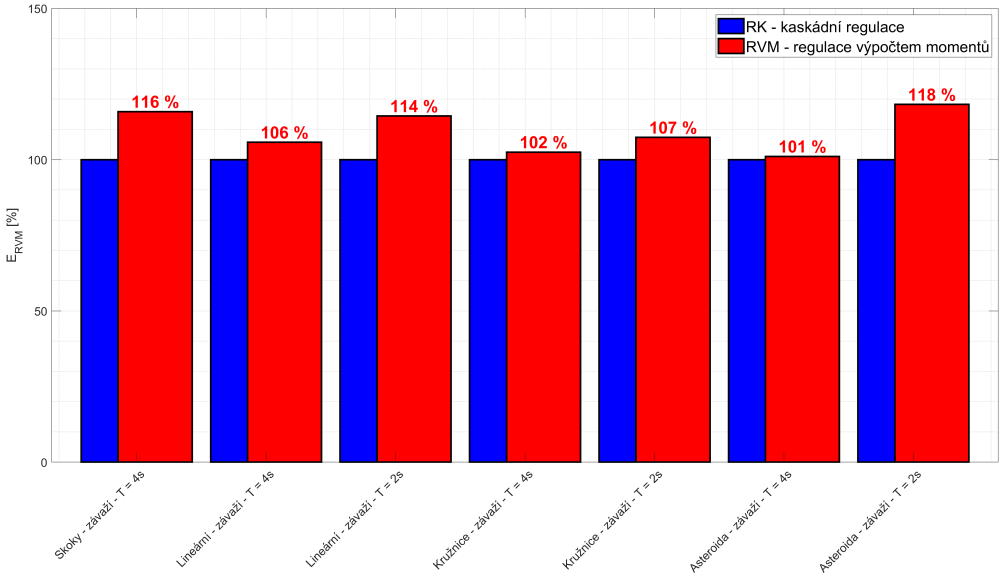
4.9.3 Porovnání z pohledu nároků na elektrickou energii

Mezi další metody vyhodnocení bylo zařazeno porovnání z pohledu energetické náročnosti jednotlivých typů regulátorů. Pro reálnou spotřebu elektrické energie by bylo potřeba kromě proudu měřit i napětí, což servozesilovač nedovoluje, nebo připojit na vstup do systému wattmetr. Zde půjde však pouze o poměrové porovnání mezi jednotlivými metodami. Při uvažování stejné rezistance při obou metodách regulace na stejných testovacích trajektoriích a měření se stejnou časovou diferencí (stejná perioda vzorkování), půjde tedy pouze o poměr sumy kvadrátů proudů.



Obr. 4.139 Porovnání RK a RVM z pohledu energetických nároků

Na Obr. 4.139 jsou vykresleny procentuální energetické nároky RVM oproti normovaným nárokům RK. Je vidět, že ve všech případech je RVM energeticky náročnější, pro nejdynamičtější lineární a cykloidní průběh dokonce více než 3-násobně. Se zvyšující se požadovanou dynamikou průběhů roste i rozdíl mezi využitou energií obou regulátorů.



Obr. 4.140 Porovnání RK a RVM z pohledu energetických nároků - závaží

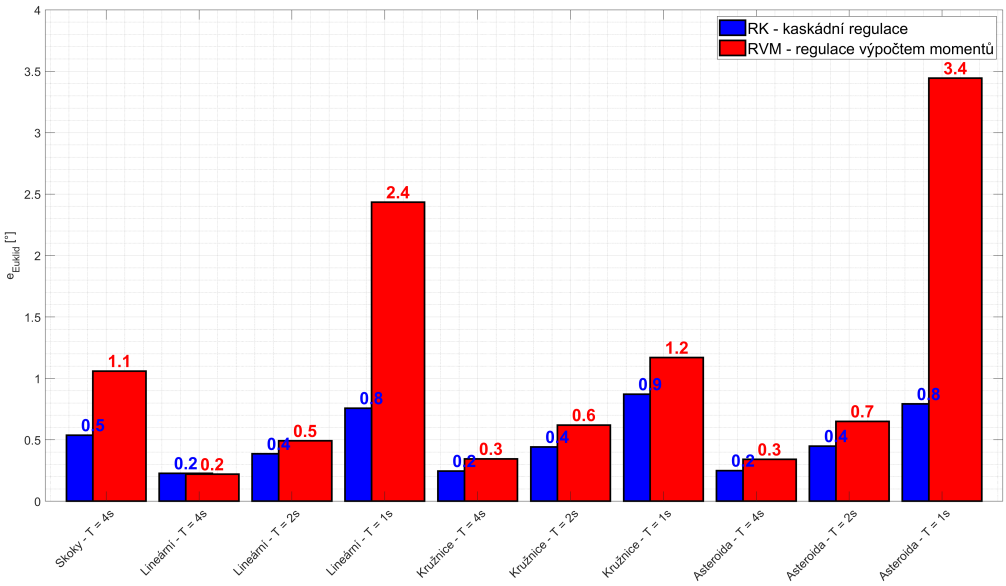
V případě trvalé poruchy v podobě závaží se již energetické nároky vyrovnávají, pro RVM však stále převyšují RK - Obr. 4.140. Je to způsobeno proudovou saturací, do které se v tomto případě dostává i RK, která bez závaží dosáhla proudového limitu jen zřídka.

Pro standardní situaci bez závaží vyžaduje RVM průměrně o cca 70% více elektrické energie, než běžná RK. Vše však záleží na nastavení regulátorů, uvedený příklad platí pouze pro zde probíranou konfiguraci.

4.9.4 Porovnání podobnosti chování reálného a fyzikálního modelu

Tato metoda vyhodnocuje podobnost chování fyzikálního a reálného modelu při regulaci pomocí RK a RVM. Tím lze určit míru důvěryhodnosti nastaveného regulátoru v simulaci, tedy míru pravděpodobnosti, že bude nastavený regulátor v simulaci fungovat i na reálném systému. Na Obr. 4.141 jsou zobrazeny průměrné Euklidovské odchylky pro jednotlivé testovací trajektorie při regulaci pomocí RK a RVM. Je vidět, že až na jeden případ dosahuje RK lepší podobnosti simulace s reálným systémem, než RVM. Při zvyšující se dynamice pohybu klesá u obou metod podobnost se simulací, protože se začnou projevovat vlivy, které v simu-

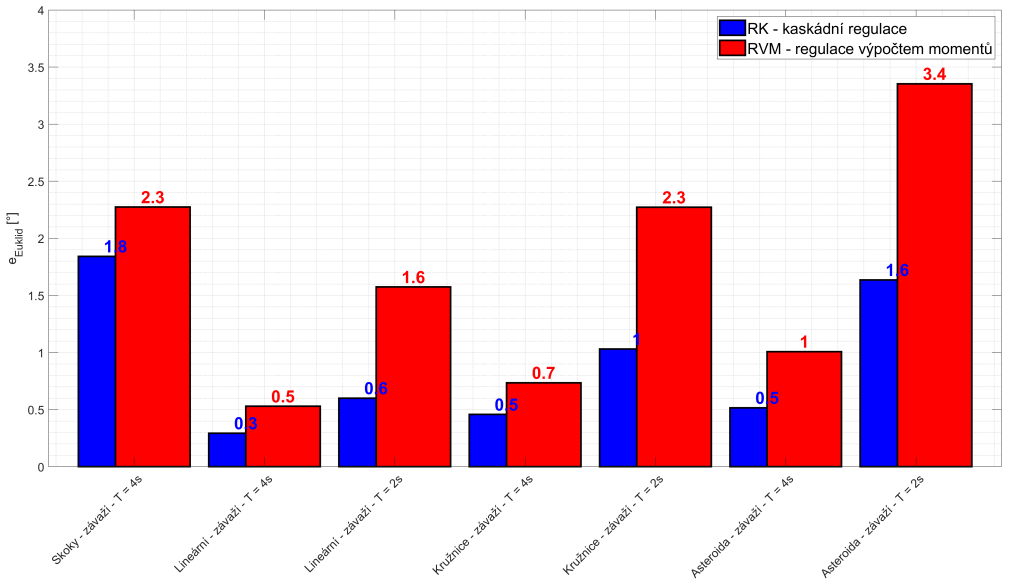
laci nebyly uvažovány. U RK i RVM klesá míra podobnosti v podstatě stejně, pokud není dynamika pohybu příliš ovlivněna proudovým limitem, což je vidět u lineárního a cykloidního průběhu s $T = 1s$.



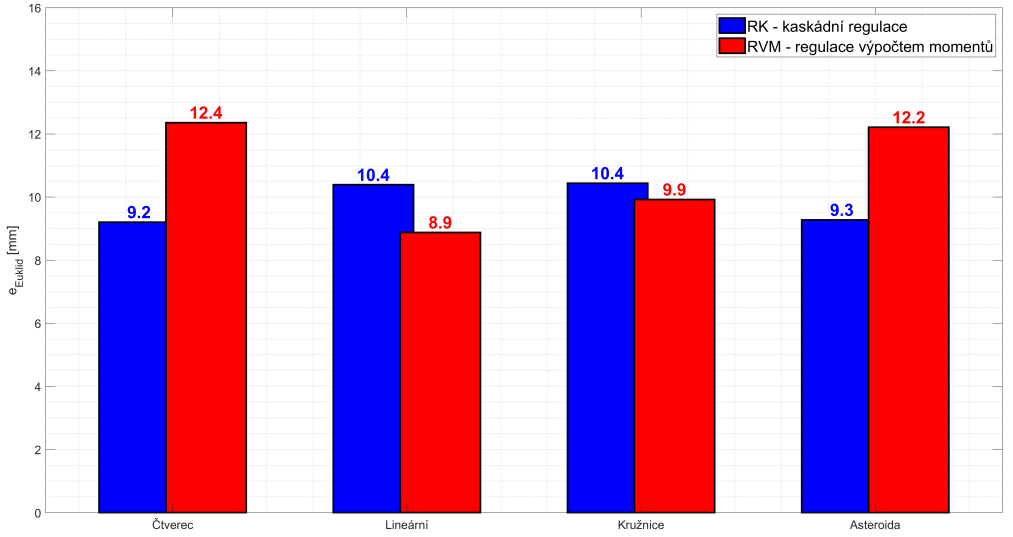
Obr. 4.141 Porovnání RK a RVM z pohledu podobnosti simulačního a reálného chování systému při regulaci roviny

Obr. 4.142 demonstruje odchylku reálného měření od simulace pro případ se závažím. Zde je podobnost ještě menší, zejména kvůli pružnosti vyvolané vyšší setrvačností, což platí hlavně pro RVM, který má obecně ostřejší projevy a tím je zde i vyšší vliv elasticity materiálu.

Podobnost průběhů při regulaci kuličky je zobrazena na Obr. 4.143. Zde jsou výsledky v podstatě totožné, pro *Čtverec* a *Asteroidu* dominuje v podobnosti reálného systému a simulace RK, pro *Lineární* a *Kružnici* zase RVM. Průměrná Euklidovská odchylka mezi průběhy se pohybuje okolo 10 mm a pro všechny typy testovacích trajektorií se liší minimálně.



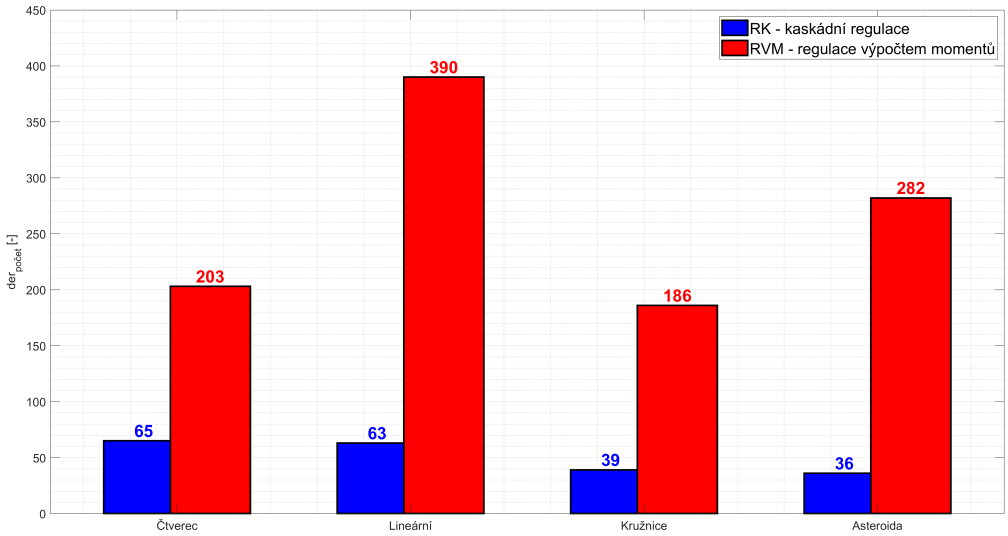
Obr. 4.142 Porovnání RK a RVM z pohledu podobnosti simulačního a reálného chování systému při regulaci roviny - závaží



Obr. 4.143 Porovnání RK a RVM z pohledu podobnosti simulačního a reálného chování systému při regulaci kuličky

4.9.5 Porovnání mechanického zatížení systému

Poslední porovnávací metodou je vyhodnocení mechanického zatížení systému při regulaci RK a RVM. Uvažované mechanické zatížení testované soustavy vychází zejména z kmitání nakloněné roviny. Posuzovacím kritériem je tedy změna směru rychlosti náklonu roviny, protože se při ní projevují vůle systému, dochází k opotřebení ozubených soukolí a ve výsledku ke stále se zvětšující hysterezní smyčce. Také tím trpí uchycení výstupů převodovek, které se v přechodu kov/-plast vymačkávají, vlivem pružností a změnami směru rychlostí dochází k namáhání materiálů a kmitání roviny až v okolí vlastních frekvencí, což vede k zesílení kmitání a rozkmitu celé sestavy apod.

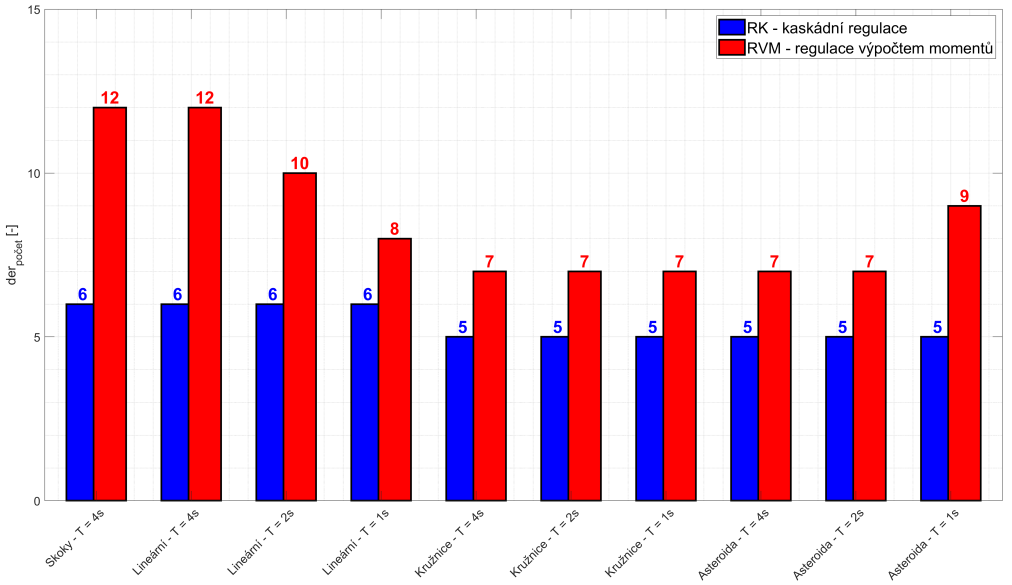


Obr. 4.144 Porovnání RK a RVM z pohledu mechanického zatížení systému při regulaci kuličky

Mechanické zatížení bude tedy vyhodnoceno jako počet změn znaménka derivace měřeného náklonu roviny, na všech testovacích trajektoriích. Z pozorování vyplynulo, že stěžejním bodem zájmu jsou vůle v ozubeném soukolí, které způsobují mnohokrát zmiňovanou rezonanci systému. Aby nebylo měření ovlivněno šumem a byly vyhodnocovány opravdu mechanické rázy v použité převodovce, byl stanoven mezní úhel na $5arcmin$. Změna znaménka derivace pod touto mezí není zahrnována do porovnání, protože ještě nedochází k mechanickému rozpo-

jení styčných ploch zubů. Mezní úhel je určen jak pomocí tabulkových hodnot použité převodovky, tak reálným ověřením.

Na Obr. 4.144 je vidět to, co bylo prezentováno již na Obr. 4.132. Metoda RVM dosahuje cca 3 až 8 krát vyššího mechanického zatížení, a to podle typu žádané trajektorie. Nejmenší rozdíl mezi metodou RK a RVM jsou u skokové změny žádané hodnoty kuličky ve tvaru čtverce, největší zase u cykloidního průběhu. Je nutné podotknout, že graf na Obr. 4.144 je výsledkem vyhodnocení náklonů roviny při regulaci kuličky, tedy při rychle se měnících skokových změnách žádané hodnoty RK a RVM.



Obr. 4.145 Porovnání RK a RVM z pohledu mechanického zatížení systému při regulaci roviny

Při čistém porovnání RK a RVM bez nadřazené regulační smyčky kuličky, jsou už rozdíly mezi oběma metodami méně patrné. RVM sice obsahuje ve všech případech stále více mechanických rázů, ale maximálně dvojnásobek oproti RK - viz. Obr. 4.145.

4.9.6 Shrnutí

Po vyhodnocení všech metod porovnání lze tvrdit, že:

- Z pohledu **Euklidovské regulační odchylky** při regulaci roviny je RVM jednoznačně vhodnější, RK ztrácelo zejména kvůli trvalé regulační odchylce při trackingu. Při řízení kuličky byla RK i RVM v podstatě rovnocenná, mírně lepších výsledků dosahovala RK.
- Při porovnání **rychlosti regulace** se obě metody lišily průměrně o 20% ve prospěch RVM.
- Co se týče **energetických nároků**, je ve všech případech RVM náročnější. Při nízkých dynamikách pohybu převyšuje RVM energetickou náročnost RK pouze v desítkách procent, u průběhů s vysokou dynamikou se vyšplhá až na trojnásobek.
- U vyhodnocení **podobnosti chování simulace a reálného systému** dosahuje RK menších odchylek a je tedy větší pravděpodobnost, že nastavený regulátor v simulaci bude fungovat velice podobně i v reálných podmínkách. U RVM jsou větší odchylky zejména při vysoké dynamice pohybu, při nízkých rychlostech se RK a RVM liší pouze v desítkách procent.
- Z pohledu **mechanického zatížení** je zřejmé, že dosahuje vyšších hodnot RVM. Významný rozdíl je viditelný hlavně při použití regulátoru s regulací kuličky, u řízení samotné roviny není rozdíl tak znatelný.

Po vyhodnocení a srovnání obou regulátorů nelze jednoznačně říci, jaký typ je lepší, nebo horší. Každý má zkrátka svoje pozitiva i negativa a je nutné volit vždy podle způsobu použití na konkrétním systému a s konkrétními požadavky na průběh regulace.

5 HLAVNÍ VÝSLEDKY PRÁCE

Tato dizertační práce se zaměřuje na vyhodnocení a porovnání moderních metod řízení pohybových stavů mechanické soustavy, jejíž dynamické projevy odpovídají chování sériového průmyslového robotu. Protože ověření na dostupném reálném průmyslovém robotu není v současnosti možné (nutný přístup alespoň do proudové smyčky robotu, což žádný výrobce nedovoluje), bylo nutné postavit vlastní robotickou strukturu, která má k dispozici detailní matematický a fyzikální model pro analýzu a syntézu, prvky moderních přístupů k řízení pohybu a je plně otevřená pro vývoj. Právě tato výsledná komplexní mechatronická struktura se všemi rozšířeními je hlavním výsledkem práce, spolu s dvěma navrženými, otestovanými, porovnanými a v samotné podstatě řízení se lišícími metodami regulace. Níže jsou do podkapitol rozděleny dílčí výsledky.

5.1 Reálně sestavená robotická struktura

Reálný model, sloužící pro testování teoreticky funkčních strategií řízení pohybu, je pokládán za stěžejní část této práce. Bez možnosti prakticky ověřit použitelnost navržených regulátorů, by se všechna tvrzení zakládala pouze na porovnání s matematickým, v lepším případě fyzikálním, modelem. To, že se ale reálný systém chová stejně jako matematický/fyzikální model, je však pouhý předpoklad, než dojde ke vzájemnému porovnání. Správnost a přesnost matematického/fyzikálního modelu je založena, ať už více či méně, na lidském faktoru, a zanedbání/přehlédnutí důležitých vlastností se zjistí až při porovnání s fyzikální realitou. To bylo mnohokrát potvrzeno v průběhu celé práce, kdy na „papíře“ i v simulaci vše perfektně sedělo a fungovalo, než došlo na implementaci do reálného prostředí. Fyzikální limity a různé zanedbané omezení, které jsou na první pohled nepodstatné, se v synergickém efektu mohou promítnout jako nezanedbatelný problém, který je nutný nějakým způsobem vyřešit. Představovaná robotická struktura (Obr. 4.17 a 4.18) je výsledkem takovéhoto řešení a postupně procházela různými úpravami (Obr. 4.15 a 4.16). Obsahuje aktivně i pasivně chlazenou desku s ovládacím panelem a elektronikou umístěnou v transparentním stole, otočný dotykový displej s gyroskopickou myší, hrazdu s přísvitem a kamerou a samotný řízený systém v podobě nakloněné roviny s 2 přímo řízenými 2 nepřímo

řízenými zobecněnými stupni volnosti. Většina dílů je vyrobená technologií 3D tisku z materiálu ABS a PETG, ostatní prvky jsou opracovány soustružením, frézováním, svařováním, vrtáním, broušením, leštěním apod. Na typovém uspořádání, konstrukčním návrhu, výrobě jednotlivých komponent, elektrickém zapojení a fyzické, logické ani signálové topologii se nepodílel nikdo další a je výhradně autorovou prací. Sestavení celého reálného systému je popisováno v kapitolách 4.1 až 4.4.

5.2 CAD model

Všechny díly mají svůj CAD model a jsou tedy modifikovatelné ve vztahu ke zbytku celé sestavy. Dílům, které nebyly přímo vyrobené (napájecí zdroje, servozesilovač, motory, převodovky atd.), byly vytvořeny zjednodušené modely s přesnými rozměry a fyzikálními vlastnostmi, pro potřeby fyzikální/matematické analýzy/syntézy. Kompletní CAD model (Obr. 4.4) prošel stejně jako reálný systém mnoha úpravami (Obr. 4.1 a 4.2) a ve své aktuální podobě obsahuje přes 270 dílů, tedy samostatně vymodelovaných prvků, poskládaných do jedné plně vyvazbené pohyblivé sestavy. CAD model je nedílnou součástí celé práce a je využíván od samotného návrhu typového uspořádání, až po konečný návrh zákona řízení. Bez možnosti jeho využití by bylo vyřešení některých části práce velmi komplikované, pokud by byly vůbec řešitelné (např. kapitola 4.1, 4.2, 4.7, 4.8.1, 4.8.2, 4.8.5 atd.).

5.3 Matematický model

S pomocí CAD modelu a odvozených kinematických transformačních matic v kapitole 4.7.4 byl sestavený matematické model (kapitola 4.8.1), který se následně využíval pro obě metody regulace roviny (kapitola 4.8.3 a 4.8.5). Jeho výsledného tvaru bylo dosaženo dvěma postupy, a na dynamických charakteristikách je dokázáno, že kvalifikovaným zjednodušením je možné výrazně zredukovat složitost matematického zápisu. Zjednodušený popis se využíval pro další analýzu a syntézu, přičemž se stále kontrolovalo, jestli nedosahuje přesný původní tvar lepších výsledků (např. PŘÍLOHA Q.e,f). Všechny tvary matematických modelů byly vzájemně porovnány, z čehož vyplynulo několik zajímavých závěrů, postřehů a

doporučení. Také byl provedený rozbor výsledných pohybových rovnic s výčtem vlastností řízeného systému vyplývajících z matematického popisu jeho chování.

5.4 Fyzikální model

Opět s využitím CAD modelu, v tomto případě jako základem pro kompletní fyzikální popis všech hmotných dílů systému, byl sestavený fyzikální model, popsaný v kapitole 4.8.2. Chování uvolněného objektu s 6DOF (kulička) pro konkrétní uspořádání, včetně reakcí na podložku i ohraničení roviny, je popsáno v PŘÍLOZE F a bylo ověřeno na chování reálného systému. Fyzikální model sloužil pro kontrolu správnosti odvození matematického modelu, k výběru vhodných akčních členů, představě o chování reálného systému a především jako simulační model pro nastavení a odladění všech regulátorů, kontrolu správnosti jejich přepisu do jazyka C++ a průběžné ověřování dílčích částí regulačních pochodů.

5.5 Řídicí aplikace s GUI

Pro jednoduchou parametrizaci celého systému, vizuální zpětnou vazbu a přívětivé uživatelské ovládání byla vytvořena grafická aplikace v jazyce C++, běžící na minipočítači Raspberry Pi. Obsahuje hlavní okno se zpětnou vazbou z kamery s vykreslenými výsledky algoritmů zpracování obrazu, záložky s nastavením kamery, strojového viděni, vzájemné komunikace, parametrizací a volbou typů regulátorů, ručním ovládáním, grafickým vykreslením průběhů a kalibrací. Aplikace je multiplatformní a použitý řídicí počítač obstarává zpracování obrazu, výpočet regulátorů, komunikaci mezi servozesilovačem, obsluhu grafického rozhraní a průběžný záznam dat. Vším výše popsaným se zabývá kapitola 4.5, a to od samotné struktury programu až po představení celé GUI aplikace. Program je plně otevřený a připravený na mnohá rozšíření, které mohou na tuto práci navazovat jak v Ph.D. úrovni, tak v Ing. i Bc. obtížnosti, což dokazují úspěšně obhájené studentské práce [83, 15], které řešily zde neprezentované dílčí části.

5.6 Strojové vidění

Systém využívá moderní a masivně se rozšiřující metodu strojového vidění vyhodnocením optické informace z kamery. Stejně jako u řídicího počítače byl zde kladen důraz na levné a otevřené řešení, které nepodléhá žádným licencím a je čistě na implementátorovi, jakou cestou zpracování obrazu se bude ubírat. Vyhodnocení obrazové informace využívá pouze základních funkcí a knihovny OpenCV, nejsou požity žádné hotové řešení, vše je složeno z dílčích operací s obrazem a vzhledem k výpočetnímu výkonu použitého řídicího počítače je kladen důraz na maximální optimalizaci výpočtů pro práci v reálném čase. Výpočetní čas celého strojového vidění se pohybuje v řádech jednotek ms i se zahrnutím dříve zmíněných kinematických transformací, což je vzhledem k použitému hardwaru velmi dobrý výsledek.

Stěžejní a bezesporu nejkomplikovanější částí byla rekonstrukce skutečné polohy kuličky na nakloněné rovině, s využitím kombinace matice homogenní transformace, trigonometrie, geometrických vlastností optiky kamery, výsledků kalibrační procedury a pozice kuličky ve snímku z kamery, za účelem obecné transformace z plošných souřadnic kuličky v obrazu, do prostorových souřadnic řízeného systému, a poté do 2D souřadné soustavy nakloněné roviny. Podobné transformace byly použity i pro veškeré vykreslování do snímku (ohraničení roviny, souřadná soustava, podkladová mřížka, žádaná trajektorie apod.) a využívají CAD model a stejnou obecnou kinematickou transformační matici, použitou při odvození matematického modelu. Popis strojového vidění je rozebírán v kapitolách 4.6 a 4.7, ve kterých jsou všechny části podrobně popsané a nutno přiznat, že zabíhá možná až příliš do detailů. Úroveň detailnosti a podrobnosti popisovaných problémů a jejich řešení neslouží k představení odvedené práce, ale k podání kompletních informací čtenáři, který by chtěl podobný problém řešit, nebo na něj v této práci navazovat, protože má sloužit zejména pro akademické/vědecké účely.

5.7 Regulace roviny kaskádou P(I)(D) regulátorů

Jako etalon pro porovnávání různých metod řízení pohybu byla zvolena kaskáda složená z P(I)(D) regulátorů. Její nastavení a ladění probíhalo po celou dobu ře-

šení práce od doby, kdy byl reálný model postavený a schopný řízení v uzavřené smyčce. Stávající nastavení je tedy prověřeno léty testování a lze tvrdit, že pro tuto konkrétní sestavu a požadavky na řízení je to nejlepší možné, a každá změna parametrů je krokem k celkově horšímu chování systému. Nastavená proudová smyčka je ponechána pro všechny ostatní metody řízení, protože její chování není ovlivněno připojenou zátěží. Její zápis je implementován do servozesilovače (řídicí jednotky motorů) a řídicí počítač (Raspberry Pi) posílá pouze požadavek na proud. Rychlostní a polohová smyčka má standardní strukturu, tedy PI regulátor s omezením integrační složky a filtrem 1. řádu v rychlosti a P regulátor v poloze. Postup odvození a nastavení je popisován v kapitole 4.8.3. K výpočtu parametrů regulátorů bylo využito matematického modelu (pouze moment setrvačnosti) a simulace chování probíhala nejdříve na fyzikálním modelu, až poté ma reálném systému. Výsledky jsou více než uspokojivé, regulátor je robustní a minimálně náchylný na změny parametrů soustavy, chod systému je plynulý a bez mechanických rázů. Problémy nastávají zejména při zvýšení periody vzorkování nad 10 ms, což by mohl být u systému bez reálného času problém. Tento jev byl však pozorován až při úmyslném zvýšení periody vzorkování, za provozu je dosahováno průměrné hodnoty pod 1 ms.

5.8 Regulace roviny výpočtem točivých momentů

Jako jeden z moderních přístupů k řízení pohybu byla vybrána metoda výpočtu točivých momentů. Jedná se o zcela odlišnou koncepci oproti kaskádě P(I)(D) regulátorů a celé odvození, nastavení a ověření funkčnosti je popisováno v kapitole 4.8.5. Pro reálné použití této metody bylo nutné udělat rozvahu nad možným přístupem, která zahrnuje použití průmyslové/filtrované derivace s ověřením jejího chování a přínosů, eliminaci šumu snímaného signálu, filtraci žádaných hodnot, doplnění o automaticky připínatelnou integrační složku s omezeným rozsahem a zahrnutí reálné periody vzorkování i do simulačního modelu. Nastavení regulátoru probíhalo samozřejmě nejdříve v simulaci s využitím typových testovacích trajektorií, a celý postup byl krok po kroku popsán a zdůvodněn. Testy na reálném systému byly z počátku problematické, protože zanedbání na první pohled nepodstatných odlišností mezi reálným a fyzikálním modelem, vedly k velkým

odlišnostem v chování při regulaci. Až popisovaná rozvaha nad možným přístupem dopomohla k úspěšnému zprovoznění tohoto typu regulátoru na reálném systému, a postupně byly regulační pochody doladěny do uspokojujícího stavu. Tato metoda není tak robustní jako P(I)(D) regulace, její nastavení v reálném systému je obtížnější a je vhodné maximálně minimalizovat odlišnosti mezi simulací a realitou. Chování RVM při regulaci je velice agresivní a podstatně rychleji reaguje na změnu žádané hodnoty, než RK. Ve vyhodnocení dosahovala RVM ve většině zvolených kritérií lepších výsledků než RK, vždy je ale nutné zvolit vhodnou metodu v kontextu požadavků, které jsou na systém kladeny.

5.9 Regulace kuličky

Regulace kuličky je doplňující možností celého systému, protože neslouží k hlavní části práce, kterou je porovnání standardního a moderního přístupu k řízení pohybu. Je však nezanedbatelnou součástí, s kterou je již od začátku počítáno a která výrazně rozšiřuje použitelnost celého systému, jehož podoba se přizpůsobovala právě této možnosti. Úkoly související s regulací kuličky zabírají minimálně čtvrtinu veškeré probírané problematiky a dávají systému potenciál pro budoucí rozšíření v oblasti pokročilých metod zpracování obrazu s využitím adaptivních přístupů, neuronových sítí, pro testování řešení souvisejících se zpracováním obrazu na reálných datech, s využitím výsledků přímo pro účely řízení pohybu apod. Pro samotnou regulaci kuličky byla zvolena pouze základní metoda řízení v podobě PD regulátoru, rozšířeného o automaticky připínatelnou integrační složku s omezením maximální hodnoty a rekonstruovanou zpětnovazební rychlostí pomocí průmyslové/filtrované derivace. Tato metoda má demonstrovat, že systém jako celek funguje a je schopný využít podřízených smyček pro řízení polohy kuličky. Nastavení, ladění a ověření funkčnosti na různých testovacích trajektoriích je probíráno v kapitole 4.8.4 a opět bylo chování nejdříve odsimulováno, až poté testováno na reálném systému. Je dosaženo dobrých výsledků, a i vzhledem k charakteru chování uvolněného objektu lze říci, že při zvolení vhodné podřízené řídicí struktury je systém robustní i při nízké vzorkovací frekvenci, kterou dovoluje použité technické vybavení. Je nutné podotknout, že hlavním faktorem omezujícím periodu vzorkování je čas sejmutí snímku z kamery (cca 20

ms). Jakmile je snímek dostupný, veškeré obrazové operace, složité kinematické transformace, vykreslování obrazu, výpočet regulátoru, komunikace mezi servozesilovačem a mnoho dalších, trvá již v řádu jednotek ms (cca 5 ms). Při zvýšení vzorkovací frekvence by dosahovala regulace kuličky podstatně lepších výsledků.

5.10 Závěrečné vyhodnocení a porovnání

Před souhrnným srovnáním byly porovnány regulační pochody RVM a RK na testovacích trajektoriích typu skokové změny, lineární změny, harmonické změny ve tvaru kružnice a neharmonické změny ve tvaru asteroidy. Porovnání probíhalo jak při běžných podmínkách, tak při trvalé poruše v podobě přidaného závaží, a pro několik stupňů žádané dynamiky pohybu. Vše je popisováno v kapitole 4.8.7. Zde jednoznačně dominuje RVM, protože to co je na první pohled vidět, je regulační odchylka. Jako doplňující část bylo uvedeno taktéž porovnání regulátorů při regulaci kuličky a je rozebíráno v kapitole 4.8.8. Zde se rozdíl mezi RVM a RK v podstatě ztrácí a regulátory jsou rovnocenné, důvody a méně patrné rozdíly jsou zde opět popsány a vysvětleny.

K závěrečnému vyhodnocení obou regulátorů bylo stanoveno celkem 5 kritérií, které zahrnují široké spektrum různých pohledů na kvalitu regulace. Patří mezi ně kritérium Euklidovké regulační odchylky, rychlost regulace, energetické nároky, podobnost chování v simulaci a v reálných podmínkách a rozdíly v mechanickém zatížení systému. Porovnání je znázorněno na přehledných sloupcových grafech s popisem a vysvětlením. Většina závěrů se vztahuje na regulaci roviny, kde je to ale možné, jsou uvedeny i případy pro regulaci kuličky. Všechny hodnotící kritéria jsou na závěr shrnuty s poznámkou, jaký regulátor dosáhl pro danou metodu porovnání lepších výsledků.

6 PŘÍNOS PRÁCE PRO VĚDU A PRAXI

Využití moderních metod řízení pohybu s nástupem počítačů se stále se zvyšujícím výpočetním výkonem je aktuálním tématem, kterým se začíná zabývat i průmyslové odvětví. Výrobci robotických systémů se předhánějí v parametrech svých produktů, mezi které patří zejména rychlost, přesnost, životnost a cena celé sestavy. Se stávajícím autonomním systémem řízení jsou průmyslové roboty nyní na svých limitech, kvůli robustnosti obsahují předimenzované akční členy a jsou celkově velmi zatlumené. Řešením by bylo využít jiný způsob řízení motorů v kloubech, neautonomní MIMO regulátory, které mají informace o ostatních dějích v systému a jsou schopny pružně reagovat na vzniklé požadavky řízení. Právě takovým je např. regulátor s výpočtem momentů, který byl v této práci vybrán za představitele jedné z moderních metod řízení pohybu a je porovnán s klasickým přístupem k regulaci.

Vědecký přínos práce je jak v teoretické části, která obsahuje množství nestandardních přístupů k řešení zadaných technických problémů, tak především v části praktické, v podobě reálně postaveného systému, který je plně otevřený a využitelný i pro testy jiných, teoreticky funkčních návrhů, které je potřebné ověřit i z praktického hlediska. Systém má jak možnost testování různých přístupů k řízení pohybu, tak schopnost implementace a ověření knihovních, nebo vlastních funkcí zpracování obrazu. Výsledky kamerového vyhodnocení mohou, ale nemusí, být využity přímo na řízení pohybu v reálném čase. Toto přímé a zcela vývojářsky otevřené propojení programátorské a automatizérské části na jednom řídicím počítači, kterým může být v podstatě jakýkoliv připojený počítač s operačním systémem, je z vědecko-akademického pohledu velmi přínosné a pohodlné. To platí zejména v okamžiku, kdy je teoreticky ověřený a funkční návrh nutné aplikovat do praxe.

Praktickým přínosem je tedy propojení vědeckého/akademického řešení s praxí - možnost dokázání, že je navrhovaný přístup reálně použitelný i na skutečném mechatronickém systému. Vědecky zpracovaná část má většinou dobře zvládnutou teoretickou stránku věci, výsledky má ověřené simulačně a předpokládá, že simulační model je dostatečně přesný na to, aby vše fungovalo uspokojivě i na reálném systému. To však nemusí nutně platit, a proto se přistupuje k výrobě

prototypů, na kterých se ověřuje aplikovatelnost navrhovaného přístupu alespoň na typových úlohách, než se přistoupí k vývoji finálního produktu. Návrh, výroba, zprovoznění, obsluha a kvalifikované vyhodnocení výsledků funkčnosti takového prototypu je však zdlouhavé a mnohdy časově nejnáročnější částí celé práce, což bylo i v tomto případě nejednou ověřeno. Sestavené reálné zařízení by tedy mohlo sloužit jako odladěný testovací systém, s částí připravenou pro implementaci vlastního řešení.

Navržená soustava byla po celou dobu vývoje stavěná tak, aby byla použitelná také při výuce studentů automatizace/robotiky/informatiky, protože na trhu není k dispozici žádný podobný koncept, který by byl takto otevřený pro implementaci vlastních řešení a skýtal by srovnatelné možnosti využití. Důkazem, že je struktura použitelná i pro řešení jednoduchých problémů, jsou dvě již dříve zmiňované úspěšně obhájené bakalářské práce [83, 15]. Celé zařízení je sice odladěné a plně funkční, stále je však ve vývoji a nápady pro jeho rozšíření by vystačily na desítky bakalářských/diplomových a několik dizertačních prací. Ostatní potenciální přínosy vyplývají z detailnějšího popisu v kapitole 5.

7 ZÁVĚR

Původním záměrem práce bylo otestovat různé moderní metody řízení pohybu na reálném průmyslovém robotu, který je k dispozici na FAI UTB. Od toho bylo však postupem času nutné upustit, protože v současnosti není k dispozici žádný průmyslový robot, u kterého by výrobce umožnil přístup do řídicí jednotky robota s úpravou struktury řízení. Výrobci/prodejci dokonce tají i takové informace, jako je převodový poměr použité převodovky, rozložení hmot jednotlivých ramen, parametry akčních členů, strukturu regulátorů, apod. Skutečnost je však spíše taková, že ani sami výrobci neví, jak přesně se jejich roboty chovají a jakými technickými prostředky jsou vybaveny. Řízení zajišťují většinou autonomní servosystémy dodávané jinou firmou a celá robotická struktura je pro programátora/vývojáře jen jakousi černou skříňkou, která zajistí splnění vstupních požadavků s definovanou přesností a v definovaném čase. Návrh vlastního řízení takovéhoto systému by vyžadovalo vlastní identifikaci, analýzu a syntézu, a po zprovoznění by každá případná úprava či výměna mechanických částí znamenala nemalé finanční prostředky a dlouhé čekací lhůty, pokud by to vůbec bylo možné. S přihlédnutím mimo jiného na výše uvedené, byl upraven původní záměr a začal se od základu budovat vlastní robotický/mechatronický systém, který skýtá více možností než průmyslový robot, je plně otevřený vývojáři, jsou známé veškeré jeho vlastnosti, parametry a technické vybavení a jeho dynamické projevy odpovídají chování sériového průmyslového robotu antropomorfního typu.

V první fázi se uvažovalo o postavení vlastního 6DOF robotu s antropomorfní kinematickou strukturou, ale konstrukční složitost takového zařízení se splněním odpovídajících přesností, je zcela mimo finanční možnosti dostupné studentovi doktorského studijního programu. Byla tedy zvolena struktura s 2 přímo řízenými DOF, rozšířená o další 2 zobecněné nepřímo řízené DOF, která je v literatuře známá pod názvem „Ball & Plate“ - Kulička na nakloněné rovině. Ve všech případech je však publikovaná jako paralelní struktura, která zcela postrádá vlivy vzájemného silového působení, tak charakteristického pro sériovou kinematickou strukturu. Probíraný systém je tedy zcela odlišný od zmiňovaných paralelních uspořádání, kterých jsou v literatuře stovky, a jeho finální podoba prošla mnoha změnami a úpravami. Po sestavení prvního funkčního prototypu

se vyskytlo množství konstrukčních problémů, které byly postupně vyřešeny a nyní je stávající zařízení z konstrukčního hlediska hotové.

Paralelně s konstrukcí probíhal také výběr technického vybavení. Za aktuátory byly zvoleny PMSM s řídicí jednotkou (servozesilovačem), u které je možné využít pouze výkonovou část s nastavitelným proudovým regulátorem a zbývající regulaci implementovat samostatně dle vlastního uvážení. Výběr výkonů motorů byl volen podle výsledků dynamické simulace požadovaného typového průběhu v SIMULINKu (na sestaveném fyzikálním modelu), převodovky byly z finančních důvodů vybrány planetové v ekonomické variantě. Dalším důležitým technickým vybavením byla optická kamera a řídicí počítač, který musí zároveň sloužit jak pro řízení, tak pro zpracování obrazu z kamery. Zde se zrodil nápad na použití zcela nestandardního vybavení, a tedy nejlevnějšího dostupného minipočítače bez systému reálného času, sloužícího primárně pro výuku a hobby účely, a kamery s podobnými parametry. Cena uvedeného řešení odpovídala zhruba 1 500 Kč za celek a cílem bylo dokázat, že i pro řízení takto komplexního systému s kamerovým viděním není potřeba hardware s vysokým výpočetním výkonem. Pro splněné vytyčených cílů bylo nutné implementovat množství optimalizací, ve výsledku ale funguje vše spolehlivě, i s rezervou výpočetního výkonu.

Již při samotném návrhu typového uspořádání byl vytvářen CAD model, který je využíván při řešení téměř každé části této práce, a v některých případech je zcela nezbytný. CAD model byl využitý při výběru aktuátorů, při odvozování matematického modelu, sestavování fyzikálního modelu, výrobě celého zařízení, analýze chování aj. a obsahuje všechny podstatné díly reálného systému. Na základě CAD modelu byl odvozený matematický a fyzikální model. Matematický model slouží pro účely návrhu zákona řízení, byl odvozený dvěma různými postupy, z nichž jeden vybraný byl zjednodušený na maximální možnou míru tak, aby jeho dynamika stále odpovídala skutečnému chování. Fyzikální model je využíván pro simulaci chování reálného systému, ale také např. pro kontrolu správnosti odvození matematického modelu, dříve zmiňovaný výběr aktuátorů, kontrolu přepisu regulátorů do C++, prezentaci výsledků a mnoho dalšího. Fyzikální model obsahuje všechny díly a materiálové vlastnosti CAD modelu, je rozšířen o všechna podstatná omezení reálného systému, je plně parametrizovatelný, každý díl je možné upravit a celkově dosahuje velmi vysoké přesnosti, v

porovnání s chováním reálného zařízení.

Celý reálný systém je řízen aplikací s grafickým rozhraní, která běží na více vláknech a slouží pro veškerou parametrizaci zařízení. Se servozesilovačem motorů komunikuje pomocí UDP s periodou do 1 ms, přičemž zvládá zpracovávat obraz z kamery s frekvencí cca 40 Hz, provádět výpočet regulátorů a složité kinematické transformace, zároveň komunikovat se servozesilovačem a vykreslovat výsledky obrazového zpracování spolu s podružnými informacemi o řízeném procesu. Samotné zpracování obrazu je maximálně optimalizované, při zajištění odpovídajících světelných podmínek je i robustní a v porovnání s časem snímání obrazu z kamery má řádově menší výpočetní náročnost. Pro vyhodnocení skutečné polohy kuličky z kamery bylo taktéž použito CAD modelu, protože je potřeba pomocí komplexních kinematických transformací její polohu na nakloněné rovině rekonstruovat. Kompletně celá aplikace je napsána v C++, je multiplatformní, a pokud bude spuštěna na libovolném počítači připojeném k řízenému systému, může být tento počítač využitý namísto Raspberry Pi.

Řízení systému je zajištěno dvěma typy regulátorů, výchozí je kaskádní P(I)(D) regulace, z moderních pokročilých metod řízení pohybu byla vybrána metoda výpočtu momentů. Tyto regulátory řídí náklon roviny, regulaci kuličky obstarává PD regulátor rozšířený o automaticky připínatelnou integrační složku. Nastavení a odladění regulátorů probíhalo nejdříve na simulačním modelu (fyzikální model), až poté na reálném systému. K odvození parametrů regulátoru byl využitý matematický model. Pro implementaci metody regulace výpočtem točivých momentů do reálného systému muselo navíc proběhnout předzpracování žádaných hodnot, k rekonstrukci rychlosti se využila tzv. průmyslová/filtrovaná derivace a simulační model byl rozšířen o další detaily, které vedly ke zpřesnění jeho chování a tím možnosti dokonalejšího nastavení regulátoru přímo v simulaci. Všechny probírané regulátory byly otestovány a fungují s mírnými odlišnostmi oproti simulaci i v reálném systému.

Kvalita regulace, což je pojem vyložitelný mnoha způsoby, byla vyhodnocena pomocí 5 navržených metod na 4 typech testovacích trajektoriích se 3 stupni dynamiky pohybu. Pro každou trajektorii se na reálném systému provedlo 10 měření, z nichž byl vybrán na základě statistického vyhodnocení vždy jeden dominantní průběh. Tyto průběhy byly poté vzájemně porovnány jak mezi me-

todami regulace, tak s výsledky simulace na fyzikálním modelu. Mezi kritéria kvality regulace bylo zařazeno porovnání pomocí Euklidovké regulační odchylky, rychlosti regulace, energetické náročnosti, podobnosti chování simulačního modelu a reálného zařízení a mechanického zatížení systému. K vyhodnocení regulačních pochodů bylo provedeno přes 1000 dílčích měření, které jsou prezentovány na více než 150 grafech.

Závěrem lze říci, že každá metoda regulace má svoje pozitiva i negativa a nelze jednoznačně tvrdit, že RVM je lepší než RK, když dosahovala ve většině testovacích kritérií lepších výsledků. Zejména poslední zmiňovaná metoda porovnání mechanického zatížení systému vychází jednoznačně ve prospěch RK, a při pozorování regulačního pochodu RVM na reálném systému je na první pohled patrné, že může být tento úhel pohledu rozhodující pro výběr RK. Je zkrátka otázkou priorit, jaká metoda řízení bude pro ten či onen systém vhodnější, a konečná volba bude vždy jakýmsi kompromisem mezi rychlostí a přesností regulace vs robustností a plynulostí regulačního pochodu. Celý řízený i řídicí systém je připraven na další rozšíření a vývoj jak metod řízení pohybu, tak principů strojového vidění. Nad touto prací strávil autor více než 10 000 hodin svého času, a proto věří, že budou alespoň její dílčí části využity k akademickým, vědeckým i praktickým účelům.

SEZNAM POUŽITÉ LITERATURY

[1] Aguilar-Ibáñez, C., Mendoza-Mendoza, J. and Dávila, J. Stabilization of the cart pole system: by sliding mode control. *Nonlinear Dynamics*. 2014, 78, 4, pp. 2769–2777. doi: 10.1007/s11071-014-1624-6.

[2] Ali, H. I., Jassim, H. M. and Hasan, A. F. Optimal Nonlinear Model Reference Controller Design for Ball and Plate System. *Arabian journal for science and engineering (2011)*. Nov 07, 2018, 44, 8, pp. 6757–6768. doi: 10.1007/s13369-018-3616-1. Dostupné z: <link.springer.com/article/10.1007/s13369-018-3616-1>.

[3] Alvarez-Rodríguez, S. and Lecona, F. G. P. n-th Order Sensor Output to Control k-DoF Serial Robot Arms. *Journal of Sensors*. Jan 11, 2021, 2021, pp. 1–14. doi: 10.1155/2021/8884282. Dostupné z: <dx.doi.org/10.1155/2021/8884282>.

[4] Ashi, M. M. A., Elaydi, H. and Hadrous, I. A. Modelica Based Object-Oriented Modeling and PD-Computed Torque Control of a 2-DOF Robotic Arm. pp. 13, Piscataway, Jan 1, 2018. The Institute of Electrical and Electronics Engineers, Inc. (IEEE). Dostupné z: <search.proquest.com/docview/2132078467>.

[5] Atherton, D. P. *Nonlinear Control Engineering*. Van Nostrand Reinhold Company, 1 edition, 1975.

[6] Awtar, S., Bernard, C., Boklund, N., Master, A., Ueda, D. and Craig, K. Mechatronic design of a ball-on-plate balancing system. *Mechatronics*. 2002, 12, 2, pp. 217–228. doi: 10.1016/S0957-4158(01)00062-9.

[7] Ba, D. X. and Bae, J. A Nonlinear Sliding Mode Controller of Serial Robot Manipulators With Two-Level Gain-Learning Ability. *IEEE access*. 2020, 8, pp. 189224–189235. doi: 10.1109/ACCESS.2020.3032449. Dostupné z: <ieeexplore.ieee.org/document/9233324>.

[8] Bang, H. and Lee, Y. S. Embedded Model Predictive Control for Enhancing Tracking Performance of a Ball-and-Plate System. *IEEE Access*. 2019, 7, pp. 39652–39659. doi: 10.1109/ACCESS.2019.2907111.

[9] Bang, H. and Lee, Y. S. Implementation of a Ball and Plate Control System Using Sliding Mode Control. *IEEE access*. 2018, 6, pp. 32401–32408. doi: 10.1109/ACCESS.2018.2838544. Dostupné z: <ieeexplore.ieee.org/document/8361429>.

[10] Barth, A., Weise, C. and Reger, J. Application of Higher-Order Sliding-Modes to a Ball and Plate System. pp. 192–197. IEEE, Jul 2018. doi: 10.1109/VSS.2018.8460469. Dostupné z: <ieeexplore.ieee.org/document/8460469>.

[11] Bdoor, S. R., Ismail, O., Roman, M. R. and Hendawi, Y. Design and implementation of a vision-based control for a ball and plate system. pp. 1–4. IEEE, 2016. doi: 10.1109/ICIEAM.2016.7910965. Dostupné z: <ieeexplore.ieee.org/document/7910965>.

[12] Benabdallah, I., Bouteraa, Y., Boucetta, R. and Rekik, C. Kinect-based Computed Torque Control for lynxmotion robotic arm. pp. 1–6. University of Al Qayrawan, Tunisia, Dec 2015. doi: 10.1109/ICMIC.2015.7409416. Dostupné z: <ieeexplore.ieee.org/document/7409416>.

[13] Betancourt, F. I. R., Alarcon, S. M. B. and Velasquez, L. F. A. Fuzzy and PID controllers applied to ball and plate system. pp. 1–6. IEEE, Oct 2019. doi: 10.1109/CCAC.2019.8921113. Dostupné z: <ieeexplore.ieee.org/document/8921113>.

[14] Borah, M., Roy, P. and Roy, B. K. Enhanced Performance in Trajectory Tracking of a Ball and Plate System using Fractional Order Controller. *Journal of the Institution of Electronics and Telecommunication Engineers*. Jan 02, 2018, 64, 1, pp. 76–86. doi: 10.1080/03772063.2017.1343157.

[15] BRABOREC, P. Integrace bezdrátového gyroskopického ovladače do reálného systému řízení pohybu s grafickou vizualizací snímaných veličin, 2020. Dostupné z: <hdl.handle.net/10563/47961>.

[16] Camacho, E. F. and Bordos, C. *Model Predictive control.* Springer-Verlag, 2 edition, 2007. ISBN 978-1-85233-694-3.

[17] Camargo, J., Machado, Almeida, E. and Silva, E. Mechanical properties of PLA-graphene filament for FDM 3D printing. *The International Journal of Advanced Manufacturing Technology.* Aug 19, 2019, 103, 5, pp. 2423–2443. doi: 10.1007/s00170-019-03532-5.

[18] Chang, K.-H. *Motion Simulation and Mechanism Design.* SDC Publications, 1 edition, 2016. ISBN 9781630570538.

[19] Dobriborsci, D., Margun, A. and Kolyubin, S. Discrete Robust Controller for Ball and Plate System. pp. 1–9. IEEE, Jun 2018. doi: 10.1109/MED.2018.8442461. Dostupné z: <ieeexplore.ieee.org/document/8442461>.

[20] Dodds, S. J. Settling time formulae for the design of control systems with linear closed loop dynamics. Jan 2008. Dostupné z: <repository.uel.ac.uk/item/86590>.

[21] Fabregas, E., Dormido-Canto, S. and Dormido, S. Virtual and Remote Laboratory with the Ball and Plate System. *IFAC-PapersOnLine.* Jul 2017, 50, 1, pp. 9132–9137. doi: 10.1016/j.ifacol.2017.08.1716.

[22] Fan, X., Zhang, N. and Teng, S. Trajectory planning and tracking of ball and plate system using hierarchical fuzzy control scheme. *Fuzzy Sets and Systems.* 2004, 144, 2, pp. 297–312. doi: 10.1016/S0165-0114(03)00135-0.

[23] Finta, B. and Kiss, B. Two-Degree s-of-Freedom Controller Synthesis to Robustify the Computed Torque Method. pp. 1–6. IEEE, Oct 15, 2020. doi: 10.1109/ISMCR51255.2020.9263772. Dostupné z: <ieeexplore.ieee.org/document/9263772>.

[24] Galvan-Colmenares, S., Moreno-Armendáriz, M. A., Jesús Rubio, J., Ortíz-Rodriguez, F., Yu, W. and Aguilar-Ibáñez, C. F. Dual PD Control Regulation with Nonlinear Compensation for a Ball and Plate System. *Mathematical Problems in Engineering.* Apr 16,

2014, 2014, pp. 1–10. doi: 10.1155/2014/894209. Dostupné z: <dx.doi.org/10.1155/2014/894209>.

[25] Haber, R., Bars, R. and Schmitz, U. *Predictive control in process engineering*. Wiley-VCH, 1 edition, 2011. ISBN 978-3-527-31492-8.

[26] Han, S., Wang, H. and Tian, Y. Integral backstepping based computed torque control for a 6 DOF arm robot. pp. 4055, Piscataway, Jan 1, 2017. The Institute of Electrical and Electronics Engineers, Inc. (IEEE). Dostupné z: <search.proquest.com/docview/1919645067>.

[27] Ho, M.-T., Rizal, Y. and Chu, L.-M. Visual Servoing Tracking Control of a Ball and Plate System: Design, Implementation and Experimental Validation. *International Journal of Advanced Robotic Systems*. Jul 09, 2013, 10, 7, pp. 287. doi: 10.5772/56525. Dostupné z: <journals.sagepub.com/doi/full/10.5772/56525>.

[28] Huang, D.-S., Bevilacqua, V., Figueroa, J. C. and Gan, Y. Advanced Intelligent Computing. In *7th International Conference, ICIC 2011, Zhengzhou, China*, pp. 538–543, Berlin, August 11-14, 2011 November 7, 2011. Heidelberg Springer Berlin Heidelberg 2012.

[29] Jeon, J.-H. and Hyun, C.-H. Adaptive sliding mode control of ball and plate systems for its practical application. pp. 119–123. IEEE, Apr 2017. doi: 10.1109/ICCRE.2017.7935054. Dostupné z: <ieeexplore.ieee.org/document/7935054>.

[30] Josef, P. Original Prusa i3 MK3. Dostupné z: <www.prusa3d.cz/original-prusa-i3-mk3/>.

[31] Kan, D., Xing, B., Xie, W. and Zhu, L. A minimum phase output based tracking control of ball and plate systems. *International journal of dynamics and control*. Jun 06, 2021, 10, 2, pp. 462–472. doi: 10.1007/s40435-021-00824-1. Dostupné z: <link.springer.com/article/10.1007/s40435-021-00824-1>.

[32] KASSEM, A., HADDAD, H. and ALBITAR, C. Commparison Between Different Methods of Control of Ball and Plate System with 6DOF Stewart

Platform. *IFAC-PapersOnLine*. 2015, 48, 11, pp. 47–52. doi: 10.1016/j.ifacol.2015.09.158.

[33] Kastner, A., Inga, J., Blauth, T., Kopf, F., Flad, M. and Hohmann, S. Model-Based Control of a Large-Scale Ball-on-Plate System With Experimental Validation. 1, pp. 257, Piscataway, Jan 1, 2019. The Institute of Electrical and Electronics Engineers, Inc. (IEEE). doi: 10.1109/ICMECH.2019.8722850.

[34] Ker, C. C., Lin, C. E. and Wang, R. T. Tracking and balance control of ball and plate system. *Journal of the Chinese Institute of Engineers*. Apr 01, 2007, 30, 3, pp. 459–470. doi: 10.1080/02533839.2007.9671274. Dostupné z: <www.tandfonline.com/doi/abs/10.1080/02533839.2007.9671274>.

[35] Kern, J., Urrea, C., Mendez, R. and Gonzalez, G. Development of an embedded control system by means of dsPIC applied in a 4 DOF robot. *Revista IEEE América Latina*. May 2016, 14, 5, pp. 2099–2106. doi: 10.1109/TLA.2016.7530401. Dostupné z: <ieeexplore.ieee.org/document/7530401>.

[36] Khaled, N. *Virtual Reality and Animation for MATLAB® and Simulink® Users*. Springer-Verlag, 1 edition, 2012. doi: 10.1007/978-1-4471-2330-9. ISBN 9781447123293.

[37] Khan, M. U. and Iqbal, N. Control of Robot Arm PA-10 using Camera Vision. pp. 97–103. IEEE, Jan 2007. doi: 10.1109/IBCAST.2007.4379916. Dostupné z: <ieeexplore.ieee.org/document/4379916>. ISBN 2151-1403.

[38] Khudiakova, A., Arbeiter, F., Spoerk, M., Wolfahrt, M., Godec, D. and Pinter, G. Inter-layer bonding characterisation between materials with different degrees of stiffness processed by fused filament fabrication. *Additive Manufacturing*. Aug 2019, 28, pp. 184–193. doi: 10.1016/j.addma.2019.05.006.

[39] Kolouchová, M. Morfologické operace ve zpracování obrazu, 2008.

[40] Kose, E. Controller design by using non-linear control methods for satellite chaotic system. *Electrical engineering*. Oct 12, 2016,

99, 2, pp. 763–773. doi: 10.1007/s00202-016-0450-x. Dostupné z: <link.springer.com/article/10.1007/s00202-016-0450-x>.

[41] KOSE, F., KAPLAN, K. and ERTUNC, H. M. Real Time Position and Trajectory Control of Ball and Plate System Using Different Control Techniques. pp. 1–7. IEEE, Oct 2018. doi: 10.1109/CEIT.2018.8751946. Dostupné z: <ieeexplore.ieee.org/document/8751946>.

[42] Lay, M., Thajudin, N. L. N., Hamid, Z. A. A., Rusli, A., Abdullah, M. K. and Shuib, R. K. Comparison of physical and mechanical properties of PLA, ABS and nylon 6 fabricated using fused deposition modeling and injection molding. *Composites Part B*. Nov 2019, 176, pp. 107341. doi: 10.1016/j.compositesb.2019.107341.

[43] Leonhard, W. *Control of Electrical Drives*. Springer-Verlag, 3 edition, 2001. ISBN 9783540418207.

[44] Li, K., Ping, Z., Huang, Y. and Lu, J.-G. A robust output regulation approach for trajectory tracking control of ball and plate system. pp. 617–622, Piscataway, Jul 26, 2021. Technical Committee on Control Theory, Chinese Association of Automation. doi: 10.23919/CCC52363.2021.9550278. Dostupné z: <ieeexplore.ieee.org/document/9550278>.

[45] Ma, J., Tao, H. and Huang, J. Observer integrated backstepping control for a ball and plate system. *International journal of dynamics and control*. Apr 12, 2020, 9, 1, pp. 141–148. doi: 10.1007/s40435-020-00629-8. Dostupné z: <link.springer.com/article/10.1007/s40435-020-00629-8>.

[46] Mochizuki, S. and Ichihara, H. I-PD controller design based on generalized KYP lemma for ball and plate system. pp. 2855–2860. EUCA, Jul 2013. doi: 10.23919/ECC.2013.6669269. Dostupné z: <ieeexplore.ieee.org/document/6669269>.

[47] Morales, L., Gordon, M., Camacho, O., Rosales, A. and Pozo, D. A Comparative Analysis among Different Controllers Applied to the Experimental Ball and Plate System. pp. 108–114.

IEEE, Nov 2017. doi: 10.1109/INCISCOS.2017.27. Dostupné z: <ieeexplore.ieee.org/document/8328093>.

[48] Morales, L., Camacho, O., Leica, P. and Chavez, D. Sliding-Mode Control Based on a Model Reference Applied to a Non-linear Ball and Plate System with Time Delay. pp. 1–6. IEEE, Oct 2017. doi: 10.1109/CCAC.2017.8276425. Dostupné z: <ieeexplore.ieee.org/document/8276425>.

[49] Novotny, D. W. and Lipo, T. A. *Vector Control and Dynamics of AC Drives.* Oxford University Press, 1 edition, 25 July 1996. ISBN 9780198564393.

[50] OpenCvDevTeam. Image Filtering — OpenCV 2.4.13.7 documentation, 2020. Dostupné z: <docs.opencv.org/2.4/modules/imgproc/doc/filtering.html>.

[51] Oravec, M. and Jadlovska, A. Model Predictive Control of a Ball and Plate laboratory model. pp. 165–170. IEEE, Jan 2015. doi: 10.1109/SAMI.2015.7061869. Dostupné z: <ieeexplore.ieee.org/document/7061869>.

[52] Parandian, Y., Arabshahi, H. Z., Nasr, A. and Moosavian, S. A. A. Time optimized digital image processing of ball and plate system using artificial neural network. pp. 146–151. IEEE, Oct 2015. doi: 10.1109/ICRoM.2015.7367775. Dostupné z: <ieeexplore.ieee.org/document/7367775>.

[53] Pedersen, J. *Model Based and Robust Control Techniques for Internal Combustion Engine Throttle Valves.* PhD thesis, Oct 2013. Dostupné z: <repository.uel.ac.uk/item/85vw4>.

[54] Pilch, Z., Domin, J. and Szlapa, A. The impact of vibration of the 3D printer table on the quality of print. pp. 1–6. IEEE, Sep 2015. doi: 10.1109/WZEE.2015.7394045.

[55] Piltan, F., Rahmdel, S., Mehrara, S. and Bayat, R. Sliding Mode Methodology Vs. Computed Torque Methodology Using MATLAB/SIMULINK and Their Integration into Graduate Nonlinear Control Courses. *International Journal of Engineering.* 2012, 6, 3, pp. 142–177. Dostupné z: <www.researchgate.net/publication/264084551>.

[56] Roy, P., Das, A. and Roy, B. K. Cascaded fractional order sliding mode control for trajectory control of a ball and plate system. *Transactions of the Institute of Measurement and Control*. Feb 2018, 40, 3, pp. 701–711. doi: 10.1177/0142331216663826. Dostupné z: <journals.sagepub.com/doi/full/10.1177/0142331216663826>.

[57] Russell, K., Shen, Q. and Sodhi, R. S. *Kinematics and Dynamics of Mechanical Systems:*. CRC Press, 1 edition, 2017. ISBN 9781498724937.

[58] Sabet, S. and Poursina, M. Computed torque control of fully-actuated nondeterministic multibody systems. *Multibody system dynamics*. May 29, 2017, 41, 4, pp. 347–365. doi: 10.1007/s11044-017-9577-4. Dostupné z: <link.springer.com/article/10.1007/s11044-017-9577-4>.

[59] Sancak, K. V. and Bayraktaroglu, Z. Y. Nonlinear Computed Torque Control of 6-Dof Parallel Manipulators. *International journal of control, automation, and systems*. Jun 09, 2022, 20, 7, pp. 2297–2311. doi: 10.1007/s12555-021-0198-6. Dostupné z: <link.springer.com/article/10.1007/s12555-021-0198-6>.

[60] Sharkawy, A.-N. and Koustoumpardis, P. Dynamics and Computed-Torque Control of a 2-DOF manipulator: Mathematical Analysis. *International Journal of Advanced Science and Technology*. 2019, 28, 12, pp. 201–212. Dostupné z: <hal.archives-ouvertes.fr/hal-03598924/document>.

[61] Shi, J., Liu, H. and Bajcinca, N. Robust Control of Robotic Manipulators Based on Integral Sliding Mode, -10 2008.

[62] Singh, R. and Bhushan, B. Real-time control of ball balancer using neural integrated fuzzy controller. *The Artificial intelligence review*. Sep 03, 2018, 53, 1, pp. 351–368. doi: 10.1007/s10462-018-9658-7. Dostupné z: <link.springer.com/article/10.1007/s10462-018-9658-7>.

[63] Slotine, J. J. E. *Applied nonlinear control*. 1 edition, 1991. ISBN 0130408905.

[64] Sumega, M. Riadenie platformy s dvomi stupňami voľnosti, 2017, Žilinská univerzita v Žiline.

[65] Sun, Y., Tian, W., Zhang, T., Chen, P. and Li, M. Strength and toughness enhancement in 3d printing via bioinspired tool path. *Materials & Design*. Jan 5, 2020, 185, pp. 108239. doi: 10.1016/j.matdes.2019.108239.

[66] TGDrives. TG Drives. Dostupné z: <www.tgdrives.cz/>.

[67] TheRaspberryPiFoundation. Raspberry Pi 3 Model B+, 2019. Dostupné z: <www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/>.

[68] Tudić, V., Kralj, D., Hoster, J. and Tropčić, T. Design and Implementation of a Ball-Plate Control System and Python Script for Educational Purposes in STEM Technologies. *Sensors (Basel, Switzerland)*. Feb 27, 2022, 22, 5, pp. 1875. doi: 10.3390/s22051875. Dostupné z: <www.ncbi.nlm.nih.gov/pubmed/35271021>.

[69] Walker, D. A., Hedrick, J. L. and Mirkin, C. A. Rapid, large-volume, thermally controlled 3D printing using a mobile liquid interface. *Science*. Oct 18, 2019, 366, 6463, pp. 360–364. doi: 10.1126/science.aax1562.

[70] Wang, A., Li, X., He, S., Cao, X., Jing, Y. and Chen, M. Command-Filtering-Based Adaptive Finite-Time Tracking Control for Ball and Plate System. *IFAC PapersOnLine*. 2020, 53, 2, pp. 6165–6170. doi: 10.1016/j.ifacol.2020.12.1700. Dostupné z: <dx.doi.org/10.1016/j.ifacol.2020.12.1700>.

[71] Wang, A., Li, X. and Cao, X. Backstepping-based Robust H-inf Tracking Controller Design for Ball and Plate System. pp. 523–528. Technical Committee on Control Theory, Chinese Association of Automation, Jul 2019. doi: 10.23919/ChiCC.2019.8866332. Dostupné z: <ieeexplore.ieee.org/document/8866332>.

[72] Wang, L. *Model Predictive Control System Design and Implementation Using MATLAB*. Springer, 2009. ISBN 978-1-84882-330-3.

[73] Wang, L., Zhou, X. and Hu, T. A New Computed Torque Control System with an Uncertain RBF Neural Network Controller

for a 7-DOF Robot. *Tehnički vjesnik*. Oct 01, 2020, 27, 5, pp. 1492–1500. doi: 10.17559/TV-20190906094136. Dostupné z: <search.proquest.com/docview/2461031281>.

[74] Wang, L. and Guan, S. Research on Trajectory Tracking Control for SCARA Manipulator of Tea Picking Robot. pp. 2621–2625. Technical Committee on Control Theory, Chinese Association of Automation, Jul 2020. doi: 10.23919/CCC50068.2020.9189044. Dostupné z: <ieeexplore.ieee.org/document/9189044>.

[75] Yuan, S., Liu, Z. and Li, X. Modeling and simulation of robot based on matlab/simmechanics. In *Proceedings of the 27th Chinese Control Conference, CCC*, pp. 161–165, 2008. doi: 10.1109/CHICC.2008.4604913. ISBN 9787-900719706(ISBN).

[76] Zheng, L. and Hu, R. Robust and fast visual tracking for a ball and plate control system: design, implementation and experimental verification. *Multimedia Tools and Applications*. May 30, 2019, 78, 10, pp. 13279–13295. doi: 10.1007/s11042-018-6430-6.

[77] Zheng, L. and Hu, R. Robust and fast visual tracking for a ball and plate control system: design, implementation and experimental verification. *Multimedia tools and applications*. Aug 17, 2018, 78, 10, pp. 13279–13295. doi: 10.1007/s11042-018-6430-6. Dostupné z: <link.springer.com/article/10.1007/s11042-018-6430-6>.

[78] Úředníček, Z. Unitary Theory of Direct Electromechanical Transformers. *Latest Trends on Systems*. July 17-21, 2014, 1, pp. 80–85.

[79] Úředníček, Z. *Robotics*. Tomas Bata University in Zlín, 1 edition, 2012. ISBN 9788074542237.

[80] Úředníček, Z. *Aplikovaná teorie nelineárního řízení*. 1. 2019.

[81] Spaček and Vojtěšek, J. Ball & Plate Model on ABB YuMi Robot. pp. 283–291. Springer Verlag, 2019. Dostupné z: <publikace.k.utb.cz/handle/10563/1008747>.

[82] Şen, B. and Türker, T. A Comparative Study of Medieval Begging Monks and Wandering Dervishes. FSM Scholarly Journal of Humanities and Social Sciences, 2022-06-27. doi: 10.16947/fsmia.1136427. ISBN 978-1-5386-4325-9.

[83] ŠUSTEK, M. Pokročilé algoritmy zpracování obrazu v reálném systému řízení pohybu, 2021. Dostupné z: <hdl.handle.net/10563/46209>.

SEZNAM PUBLIKACÍ AUTORA

[P.1] Zátopek, J., Úředníček, Z., Machado, J. and Sousa, J. Dynamic simulation of the CAD model in SimMechanics with multiple uses, In *Turkish Journal of Electrical Engineering and Computer Sciences.* vol. 26(3), 1278-1290, Turkey, 2018. DOI: 10.3906/elk-1712-217.

[P.2] Úředníček, Z., Zátopek, J. and Vítek, R. Mechanical educational system for automatic area observation and firing control techniques, In *3rd Conference on Innovation, Engineering and Entrepreneurship, Regional HELIX 2018.* Lecture Notes in Electrical Engineering, Portugal, 2018. DOI: 10.1007/978-3-319-91334-6_150.

[P.3] Spaček, L., Vojtěšek, J. and Zátopek, J. Collaborative Robot YuMi in ball and plate control application: Pilot study, In *7th Computer Science On-line Conference, CSOC 2018.* Advances in Intelligent Systems and Computing, Zlín, 2018. DOI: 10.1007/978-3-319-91192-2_18.

[P.4] Zátopek, J. and Úředníček, Z. Dynamic Behaviour Comparison of Three Different Mathematical Model Complexities, In *28th DAAAM International Symposium on Intelligent Manufacturing and Automation.* Annals of DAAAM and Proceedings of the International DAAAM Symposium, Croatia, 2017. DOI: 10.2507/28th.daaam.proceedings.096.

[P.5] Zátopek, J. Using the Transformation Matrices not Only to Derive the Motion Equations, In *WSEAS Transactions on Applied and Theoretical Mechanics.* vol. 11, 245-252, Greece, 2016. E-ISSN: 2224-3429.

[P.6] Zátopek, J. Various Support Software Tools Using In Simulation and Identification of a Non-linear Unstable System of Motion, In *WSEAS Transactions on Systems.* vol. 15, 321-328, Greece, 2016. E-ISSN: 2224-2678.

[P.7] Zátopek, J. Simulation and Visualisation of a Laboratory Model by Supported Software Tools Connection, In *27th International DAAAM Symposium on Intelligent Manufacturing and Auto-*

mation 2016. Annals of DAAAM and Proceedings of the International DAAAM Symposium, Bosnia and Herzegovina, 2016. DOI: 10.2507/27th.daaam.proceedings.108.

[P.8] ZÁTOPEK, J. The coordinate system transformation of a serial kinematic structures and use in the derivation of systems motion equations, In *20th International Conference on Circuits, Systems, Communications and Computers (CSCC).* MATEC Web of Conferences, Corfu Island, 2016. DOI: 10.1051/matecconf/20167602017.

[P.9] ZÁTOPEK, J. The simulation of a non-linear unstable system of motion, utilising the SolidWorks and Matlab/Simulink extended libraries/toolboxes, In *20th International Conference on Circuits, Systems, Communications and Computers (CSCC).* MATEC Web of Conferences, Corfu Island, 2016. DOI: 10.1051/matecconf/20167602016.

[P.10] ZÁTOPEK, J. Motion States of Nonlinear Unstable System Simulation, Connections between SolidWorks-Matlab/SimulinkSimMechanics-3D Animation, In *XLI. Seminář ASŘ 2016.* XLI. Seminar ASR '2016 "Instruments and Control", Ostrava, 2016. ISBN: 978-80-248-3910-3.

ODBORNÝ ŽIVOTOPIS AUTORA

Osobní údaje

Jiří Zátopek, nar. 19. 01. 1991

Kozlovice 308, 739 47 Kozlovice, ČR

zatopek@utb.cz

+420 723 747 888

Vzdělání

2015 - dosud: **Ph.D.**
Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, obor Automatické řízení a informatika, Zlín

2013 - 2015: **Ing.**
Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, obor Automatické řízení a informatika, Zlín

2010 - 2013: **Bc.**
Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, obor Informační a řídicí technologie, Zlín

2006 - 2010: **Maturita**
Střední průmyslová škola elektrotechniky a informatiky, obor Elektro-počítačové systémy, Frenštát pod Radhoštěm

Studijní stáže

05/2018 - 10/2018: **5 měsíční studijní stáž - Erasmus+**
Žilinská univerzita v Žilině, Elektrotechnická fakulta, Katedra mechatroniky a elektroniky, Slovensko

05/2017 - 08/2017: **3 měsíční studijní stáž - Erasmus+**
Universidade do Minho, School of Engineering, Mechanical Engineering Department, Portugalsko

Pracovní zkušenosti

2019 - dosud: **Akademický pracovník - asistent**
UTB ve Zlíně, Fakulta aplikované informatiky, Ústav automatizace a řídicí techniky
Robotika, mechatronika, CAD návrh pro fyzikální/matematické modelování, simulace a prototypování pomocí 3D tisku, moderní metody řízení pohybu, zpracování obrazu pro účely strojového vidění, základy mikropočítačů a programování.

2017 - 2019: **Výzkumný pracovník**
Regionální výzkumné centrum CEBIA-Tech, Zlín
Dílčí výzkumný podprogram Inteligentní výrobní systémy - práce v oblasti mechatroniky a robotiky.

2014 - 2016: **PC/PLC programátor**
Dudr Tools, s.r.o., Zlín
Vývoj plně automatizovaného zařízení na ohyb zubů pilových pásů a jeho uvedení do chodu v reálném prostředí (výběr a sestavení hardwaru, elektrotechnické/datové propojení, programování PLC, image processing na PC, komunikace přes průmyslovou sběrnici).

2012 - 2014: **C/C++ programátor**
UNIS, a.s., Brno
Softwarové řešení pro sledování objektu - využití stereo CCD kamer a termokamery, měřením vzdálenosti z obrazu (zkušenosti s algoritmy zpracování obrazu a object trackingem).

2008 - 2012: **Elektrotechnik**
ZÁVODNÝ ELEKTRO s.r.o., Lhotka
Stáž a praxe v oblasti elektrotechniky (silnoproud, kompenzace jalového výkonu, domovní elektroinstalace, přepěťová ochrana, trafostanice, NN/VN revize).

Akademická činnost

Publikace - ocenění:	XLI. Semináři ASŘ 2016 – Laureate Award 5th DAAAM International Doctoral School – Festo Price 28th DAAAM - Best Poster Presentation
Projekty:	IGA 2016, 2017, 2018, 2019, 2020 CEBIA-Tech – (NPU I) č. MSMT – 7778/2014 OP PIK Aplikace, CZ.01.1.02/0.0/0.0/19_262/0020292 OP VVV, CZ.02.2.69/0.0/0.0/16_015/0002204 OP PIK Aplikace, CZ.01.1.02/0.0/0.0/20_321/0023805 MoVI-FAI, CZ.02.2.67/0.0/0.0/16_016/0002325 OP PIK Aplikace, CZ.01.1.02/0.0/19_262/0020111 TA ČR - TREND FW01010381 OP PIK PROZAX, CZ.01.1.02/0.0/0.0/20_321/0023675
Výuka - předměty:	Mechatronické systémy Konstrukce robotů a manipulátorů Programování mikropočítačů Kinematika a dynamika mechatronických systémů Řízení pohybu Akční členy v mechatronických systémech Programování Embedded systémy s mikropočítači Programování a aplikace průmyslových robotů Robotická laboratoř Robotická pracoviště

Jazykové znalosti

Český jazyk:	Mateřský jazyk
Anglický jazyk:	B2 - středně pokročilý

Řidičský průkaz

Skupiny: A B

Osobní dovednosti

- Zkušenosti s použitím různých softwarových prostředků pro řešení zadaného technického problému.
- Schopnost propojení znalostí z matematiky, fyziky, elektrotechniky, mechatroniky, programování a následné využití pro automatizaci/robotizaci.
- Manuální zručnost, technické myšlení, kreativita, zodpovědnost, spolehlivost.

Software: Matlab, Simulink/Simscape Multibody, SolidWorks, Qt Creator/OpenCV, VS Code, TIA Portal, PrusaSlicer, Inkscape, TEXstudio, Stäubli Robotics Suite, ABB RobotStudio, MS Windows, MS Office

Jazyky: C, C++, Matlab, LaTeX, VAL-3, Rapid, SQL, ST, LD, LBD

SEZNAM PŘÍLOH

PŘÍLOHA A: MATEMATICKÝ MODEL - MATLAB SKRIPT - POSTUP 1

```
% symbolicke promenne pouzite v tranformacich a pri derivacich (derivuju jen stavove promenne, takze 1. a 2. derivace DOF)
syms Sx Sy x y bR alfa beta gama c g fiX fiY;
syms alfa_1d alfa_2d beta_1d beta_2d x_1d x_2d y_1d y_2d fiX_1d fiX_2d fiY_1d fiY_2d;

% SolidWorks parametry
% *******************************************************
% *******************************************************
% konstrukcni parametry [m] a gravitacni zrychleni [m*s^-2]
c = 0.3345;
% g = 9.81;   % kvuli vyberu pres "collect" na konci musim nechat symbolicke, abych vedel, co patri do matice G

% *******************************************************
% PRO CASTECNE OBECNE ODVOZENI MODELU - ZMENA VELIKOSTI A VAHY KULICKY
% *******************************************************
syms m3 bR;
% hmoty kazde vzajemne stacionarni casti modelu [kg]
m1 = 2.17500430; % Rameno1 + kabel_osa2 ze SolidWorks
m2 = 2.00770144; % Rameno2 ze SolidWorks
% m3 = 7900*4*pi*bR^3/3; % hmotnost kulicky vypoctena ze zadaneho polomeru

% pozice teziste v lokalni souradne soustave kazde casti [m]
r1 = [-0.02983413; 0.01371001; -0.37684365; 1]; % v LSS_final
r2 = [-0.02081070; 0.00004702; -0.05392745; 1]; % v LSS_final
r3 = [0; 0; 0; 1];  % v LSS_final

% *******************************************************
% *******************************************************

% z pohledu pohybovych rovnic nezalezi na tom, jak bude natocena kamera vuci naklonen rovice (nezalezi na uhlu gama)...ten
zbytecne zeslozituje cely vyraz a podle me by mel spravne z vyslednych rovnic vypadnout, stejne jako napr. konstanty Sx a Sy.
Protoze je ale asi rotace posunuta a pak je goniometricka funkces argumentem "gama" vsim moznym pronasobena, tak uz se pak
neda vykratit a zjednodusit a napr. figuruje uz v druhe absolutni rychlosti. No a v absolutni rychlosti prece nemuze zalezet
na tom, kde a jak si umistim globalni souradnou soustavu. Zakladni myslenkou je, ze pohybove rovnice musi vypadat stejne bez
ohledu na to, kde se rozhodnu umistit globalni souradnou soustavu a proto si ji umistim ,,vhodneji" vynulovanim "gamy". To
nemuze mit zadny vliv na dynamiku pohybu naklonen roviny, kterou pohybova rovnice popisuej, protoze je vyloucene, abych
natocenim kamery zmenil dynamiku modelu. U kinematickych prepoctu u zpracovani obrazu pochopitelne byt musi, ale pro pohybove
rovnice muzu udelat kvalifikovane zjednoduseni. Po vycisleni a dosazeni konkretnich cisel pri reseni by nemel byt rozdil ve
vysledku, ale pro obecne vyjadreni a dalsi pocitani napr. v realnem case je zbytecne, abych mel slozitou rovnici, ktera vyjde
stejne, jako zjednodusena.
% OVERENO, ZE VYSE UVEDENE PLATI - PRI VOLBE JAKEKOLI "GAMY" JE VYSLEDNY PRUBEH DYNAMICKEHO CHOVANI STEJNY...TAKZE MUZU
VYNULOVAT
gama = 0;

% *******************************************************
% *******************************************************

% TRANSFORMACNI MATICE MEZI SOURADNYMI SOUSTAVAMI
% *******************************************************
% Transformacni matice z lokalni do globalni souradne soustavy jednotlivych "clanku" spjatych s DOF
% cela transformace
T_ball = [  sin(fiY)*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + cos(fiY)*(cos(alfa)*sin(fiX)*sin(gama) -
cos(fiX)*cos(gama)*sin(beta) + cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(fiY)*(cos(beta)*cos(gama) +
sin(alfa)*sin(beta)*sin(gama)) - sin(fiY)*(cos(alfa)*sin(fiX)*sin(gama) - cos(fiX)*cos(gama)*sin(beta) +
cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(beta)*sin(alfa)*sin(fiX)*sin(gama) - cos(gama)*sin(beta)*sin(fiX) -
cos(alfa)*cos(fiX)*sin(gama),   Sx + x*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + c*sin(gama) -
bR*(cos(gama)*sin(beta) - cos(beta)*sin(alfa)*sin(gama)) - c*cos(alfa)*sin(gama) - y*cos(alfa)*sin(gama);
           cos(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) + cos(beta)*cos(fiX)*cos(gama)*sin(alfa))
- sin(fiY)*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)),   - cos(fiY)*(cos(beta)*sin(gama) -
cos(gama)*sin(alfa)*sin(beta)) - sin(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) +
cos(beta)*cos(fiX)*cos(gama)*sin(alfa)), sin(beta)*sin(fiX)*sin(gama) - cos(alfa)*cos(fiX)*cos(gama) +
cos(beta)*cos(gama)*sin(alfa)*sin(fiX),   c*cos(gama) - x*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)) - Sy +
bR*(sin(beta)*sin(gama) + cos(beta)*cos(gama)*sin(alfa)) - c*cos(alfa)*cos(gama) - y*cos(alfa)*cos(gama);
           cos(fiY)*sin(alfa)*sin(fiX) - cos(alfa)*sin(beta)*sin(fiY) - cos(alfa)*cos(beta)*cos(fiX)*cos(fiY),
cos(alfa)*cos(beta)*cos(fiX)*sin(fiY) - sin(alfa)*sin(fiX)*sin(fiY) - cos(alfa)*cos(fiY)*sin(beta),
- cos(fiX)*sin(alfa) - cos(alfa)*cos(beta)*sin(fiX),                                                    - c*sin(alfa) -
y*sin(alfa) - bR*cos(alfa)*cos(beta) - x*cos(alfa)*sin(beta);
           0,
0,
0,                                                                                                      1];
```

```
% alfa
T1 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, 0, 0, 0]);

% alfa + beta
T2 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, beta, 0, 0, 0, 0]);

% beta + x
T3 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[0, beta, x, 0, 0, 0]);

% alfa + y
T4 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, y, 0, 0]);

% ********************************************************************************************************
% ********************************************************************************************************
% HMOTNÝ BOD 1
% vektor polohy v globalni souradne soustave
P1 = T1*r1;

% vektor rychlosti v globalni souradne soustave
V1 =[diff(P1(1),alfa)*alfa_1d;
    diff(P1(2),alfa)*alfa_1d;
    diff(P1(3),alfa)*alfa_1d;
    1];

% urceni absolutni rychlosti^2
pom = V1(1)^2 + V1(2)^2 + V1(3)^2;
pom = simplify(pom);

% finalni tvar kineticke nenergie
Wk_1 = 0.5*m1*pom;
Wk_1 = simplify(Wk_1);

% potencialni energie - gravitace ve smeru globalni osy Z
Wp_1 = m1*g*P1(3);

% ************************************************
% HMOTNÝ BOD 2
% vektor polohy v globalni souradne soustave
P2 = T2*r2;

% vektor rychlosti v globalni souradne soustave
V2 =[ diff(P2(1),alfa)*alfa_1d + diff(P2(1),beta)*beta_1d;
    diff(P2(2),alfa)*alfa_1d + diff(P2(2),beta)*beta_1d;
    diff(P2(3),alfa)*alfa_1d + diff(P2(3),beta)*beta_1d;
    1];

% urceni absolutni rychlosti^2
pom = V2(1)^2 + V2(2)^2 + V2(3)^2;
pom = simplify(pom);
%
% finalni tvar kineticke nenergie
Wk_2 = 0.5*m2*pom;
Wk_2 = simplify(Wk_2);

% potencialni energie - gravitace ve smeru globalni osy Z
Wp_2 = m2*g*P2(3);

% ************************************************
% HMOTNÝ BOD 3 - osa x
% vektor polohy v globalni souradne soustave
P3 = T3*r3;

% vektor rychlosti v globalni souradne soustave
V3 =[diff(P3(1),alfa)*alfa_1d + diff(P3(1),beta)*beta_1d + diff(P3(1),x)*x_1d;
    diff(P3(2),alfa)*alfa_1d + diff(P3(2),beta)*beta_1d + diff(P3(2),x)*x_1d;
    diff(P3(3),alfa)*alfa_1d + diff(P3(3),beta)*beta_1d + diff(P3(3),x)*x_1d;
    1];

% urceni absolutni rychlosti^2
pom = V3(1)^2 + V3(2)^2 + V3(3)^2;
pom = simplify(pom);

% finalni tvar kineticke nenergie
Wk_3 = 0.5*m3*pom;
Wk_3 = simplify(Wk_3);

% potencialni energie - gravitace ve smeru globalni osy Z
Wp_3 = m3*g*P3(3);
```

```
% ************************************************
% HMOTNÝ BOD 4 - osa y
% vektor polohy v globalni souradne soustave
P4 = T4*r3;

% vektor rychlosti v globalni souradne soustave
V4 =[diff(P4(1),alfa)*alfa_1d + diff(P4(1),beta)*beta_1d + diff(P4(1),y)*y_1d;
    diff(P4(2),alfa)*alfa_1d + diff(P4(2),beta)*beta_1d + diff(P4(2),y)*y_1d;
    diff(P4(3),alfa)*alfa_1d + diff(P4(3),beta)*beta_1d + diff(P4(3),y)*y_1d;
    1];

% urceni absolutni rychlosti^2
pom = V4(1)^2 + V4(2)^2 + V4(3)^2;
pom = simplify(pom);

% finalni tvar kineticke nenergie
Wk_4 = 0.5*m3*pom;
Wk_4 = simplify(Wk_4);

% potencialni energie - gravitace ve smeru globalni osy Z
Wp_4 = m3*g*P4(3);

% CELKOVA KINETICKA ENERGIE SOUSTAVY
% ********************************************
Wk = Wk_1 + Wk_2 + Wk_3 + Wk_4;
Wk = expand(Wk);
Wk = simplify(Wk);
Wk = vpa(Wk,4);

% CELKOVA POTENCIALNI ENERGIE SOUSTAVY
% ********************************************
Wp = Wp_1 + Wp_2 + Wp_3 + Wp_4;
Wp = expand(Wp);
Wp = simplify(Wp);
Wp = vpa(Wp,4);

% LAGRANGEOVY POHYBOVÉ ROVNICE
% ********************************************
L = Wk - Wp;
% Alfa;
pom = diff(L,alfa_1d);    % derivace podle zobecnene rychlosti
pom = diff(pom,alfa)*alfa_1d + diff(pom,beta)*beta_1d + diff(pom,x)*x_1d + diff(pom,y)*y_1d + diff(pom,alfa_1d)*alfa_2d +
diff(pom,beta_1d)*beta_2d + diff(pom,x_1d)*x_2d + diff(pom,y_1d)*y_2d;   % derivace podle casu
pom1 = diff(L,alfa);  % derivace podle zobecnene souradnice
p_alfa = pom - pom1; % pohybová rovnice pro první stavovou veličinu - nezapomenout jeste, ze se to rovna !!! Q_alfa !!!
p_alfa = vpa(collect(simplify(p_alfa),alfa_2d),4);

% Beta;
pom = diff(L,beta_1d);    % derivace podle zobecnene rychlosti
pom = diff(pom,alfa)*alfa_1d + diff(pom,beta)*beta_1d + diff(pom,x)*x_1d + diff(pom,y)*y_1d + diff(pom,alfa_1d)*alfa_2d +
diff(pom,beta_1d)*beta_2d + diff(pom,x_1d)*x_2d + diff(pom,y_1d)*y_2d;   % derivace podle casu
pom1 = diff(L,beta);  % derivace podle zobecnene souradnice
p_beta = pom - pom1; % pohybová rovnice pro první stavovou veličinu - nezapomenout jeste, ze se to rovna !!! Q_beta !!!
p_beta = vpa(collect(simplify(p_beta),beta_2d),4);

% x;
pom = diff(L,x_1d);    % derivace podle zobecnene rychlosti
pom = diff(pom,alfa)*alfa_1d + diff(pom,beta)*beta_1d + diff(pom,x)*x_1d + diff(pom,y)*y_1d + diff(pom,alfa_1d)*alfa_2d +
diff(pom,beta_1d)*beta_2d + diff(pom,x_1d)*x_2d + diff(pom,y_1d)*y_2d;   % derivace podle casu
pom1 = diff(L,x);  % derivace podle zobecnene souradnice
p_x = pom - pom1; % pohybová rovnice pro první stavovou veličinu  - nezapomenout jeste, ze se to rovna !!! 0 !!!
p_x = vpa(collect(simplify(p_x),alfa_2d),4);

% y;
pom = diff(L,y_1d);    % derivace podle zobecnene rychlosti
pom = diff(pom,alfa)*alfa_1d + diff(pom,beta)*beta_1d + diff(pom,x)*x_1d + diff(pom,y)*y_1d + diff(pom,alfa_1d)*alfa_2d +
diff(pom,beta_1d)*beta_2d + diff(pom,x_1d)*x_2d + diff(pom,y_1d)*y_2d;   % derivace podle casu
pom1 = diff(L,y);  % derivace podle zobecnene souradnice
p_y = pom - pom1; % pohybová rovnice pro první stavovou veličinu  - nezapomenout jeste, ze se to rovna !!! 0 !!!
p_y = vpa(collect(simplify(p_y),alfa_2d),4);
```

```
% separace cele rovnice do jednotlivych casti - matic D, H a G - pracne, uzitecne jen pokud bych chtel separovat D, H a G,
ale
% protoze pouzivam v praci stejne automatizovany vypocet v "PohyboveRovniceFinal_postup2_6DOF", tak nechavam jen pro jistotu
% For cyklus s promennou "neco" je tady jen pro sbaleni kodu, at neprekazi
for neco = 1
% % postupny vyber prvku s vysledne pohybove rovnice do matic D, G, H
% p_alfa = vpa(collect(simplify(p_alfa),alfa_2d),4)   % D11
% alfa_2d = 0;
% p_alfa = eval(p_alfa);
% p_alfa = vpa(collect(simplify(p_alfa),beta_2d),4)   % D12
% beta_2d = 0;
% p_alfa = eval(p_alfa);
% p_alfa = vpa(collect(simplify(p_alfa),x_2d),4)      % D13
% x_2d = 0;
% p_alfa = eval(p_alfa);
% p_alfa = vpa(collect(simplify(p_alfa),y_2d),4)      % D14
% y_2d = 0;
% p_alfa = eval(p_alfa);
% p_alfa = vpa(collect(simplify(p_alfa),g),4)         % G1
% g = 0;
% p_alfa = eval(p_alfa)
% p_alfa = vpa(simplify(p_alfa),4)         % H1
%
% % a stejne tak pro vsechny ostatni pohybove rovnice, abych kompletne naplnil matice D, H a G
% p_beta = vpa(collect(simplify(p_beta),alfa_2d),4)   % D21
% alfa_2d = 0;
% p_beta = eval(p_beta);
% p_beta = vpa(collect(simplify(p_beta),beta_2d),4)   % D22
% beta_2d = 0;
% p_beta = eval(p_beta);
% p_beta = vpa(collect(simplify(p_beta),x_2d),4)      % D23
% x_2d = 0;
% p_beta = eval(p_beta);
% p_beta = vpa(collect(simplify(p_beta),y_2d),4)      % D24
% y_2d = 0;
% p_beta = eval(p_beta);
% p_beta = vpa(collect(simplify(p_beta),g),4)         % G2
% g = 0;
% p_beta = eval(p_beta)
% p_beta = vpa(simplify(p_beta),4)         % H2
%
% % ****************************************************************

% .
% .
% .
%
% g = -9.81;
% D = vpa(simplify(expand(D)),4);
% H = vpa(simplify(expand(H)),4);
% G = vpa(simplify(expand(eval(G))),4);
%
% Q = vpa(simplify(D*[alfa_2d; beta_2d; x_2d; y_2d] + H + G), 4);

end

g = -9.81;

Q_post1 = [eval(p_alfa); eval(p_beta); eval(p_x); eval(p_y)];

% pokusne vycisleni v ustalenem stavu
alfa = 0;
beta = 0;
x = 0;
y = 0;

alfa_1d = 0;
beta_1d = 0;
x_1d = 0;
y_1d = 0;

alfa_2d = 0;
beta_2d = 0;
x_2d = 0;
y_2d = 0;

vpa(eval(Q_post1),4)
```

PŘÍLOHA B: MATEMATICKÝ MODEL - MATLAB SKRIPT - POSTUP 2 - 6 DOF

```
% close all;  clear all;  clc;

% symbolicke promenne pouzite v tranformacich a pri derivacich (derivuju jen stavove promenne, takze 1. a 2. derivace DOF)
syms Sx Sy x y bR alfa beta gama c g fiX fiY par par_1d par_2d;
syms alfa_1d alfa_2d beta_1d beta_2d x_1d x_2d y_1d y_2d fiX_1d fiX_2d fiY_1d fiY_2d;

% z pohledu pohybovych rovnic nezalezi na tom, jak bude natocena kamera vuci naklonen rovice (nezalezi na uhlu gama)...ten
zbytecne zeslozituje cely vyraz a podle me by mel spravne z vyslednych rovnic vypadnout, stejne jako napr. konstanty Sx a Sy.
Protoze je ale asi rotace posunuta a pak je goniometricka funkces argumentem "gama" vsim moznym pronasobena, tak uz se pak
neda vykratit a zjednodusit a napr. figuruje uz v druhe absolutni rychlosti. No a v absolutni rychlosti prece nemuze zalezet
na tom, kde a jak si umistim globalni souradnou soustavu. Zakladni myslenkou je, ze pohybove rovnice musi vypadat stejne bez
ohledu na to, kde se rozhodnu umistit globalni souradnou soustavu a proto si ji umistim ,,vhodneji" vynulovanim "gamy". To
nemuze mit zadny vliv na dynamiku pohybu naklonen roviny, kterou pohybova rovnice popisuej, protoze je vyloucene, abych
natocenim kamery zmenil dynamiku modelu. U kinematickych prepoctu u zpracovani obrazu pochopitelne byt musi, ale pro pohybove
rovnice muzu udelat kvalifikovane zjednoduseni. Po vycisleni a dosazeni konkretnich cisel pri reseni by nemel byt rozdil ve
vysledku, ale pro obecne vyjadreni a dalsi pocitani napr. v realnem case je zbytecne, abych mel slozitou rovnici, ktera vyjde
stejne, jako zjednodusena.
% OVERENO, ZE VYSE UVEDENE PLATI - PRI VOLBE JAKEKOLI "GAMY" JE VYSLEDNY PRUBEH DYNAMICKEHO CHOVANI STEJNY...TAKZE MUZU
VYNULOVAT
gama = 0;

% pro automaticke vyhodnoceni rozmeru matic D, H, G, poctu pohybovych rovnic apod.
DOF = 6;

%%
% HODNOTY ZE SolidWorks
% *******************************************************
% PRO ODVOZENI MODELU S KONKRETNIMI CISLY
% *******************************************************
% hmoty kazde vzajemne stacionarni casti modelu [kg]
% m1 = 2.17500430; % Ramenol + kabel_osa2 ze SolidWorks
% m2 = 2.00770144; % Rameno2 ze SolidWorks
% m3 = 0.26473154; % BalancniRovina_kulicka ze SolidWorks
%
% % pozice teziste v lokalni souradne soustave kazde casti [m]
% r1 = [-0.02983413; 0.01371001; -0.37684365; 1];   % v LSS_final
% r2 = [-0.02081070; 0.00004702; -0.05392745; 1]; % v LSS_final
% r3 = [0; 0; 0; 1];  % v LSS_final
%
% % konstrukcni parametry [m]
% c = 0.3345;
% bR = 0.02;
% g = 9.81;
%%
% *******************************************************
% PRO UPLNE OBECNE ODVOZENI MODELU
% *******************************************************
% syms g c bR m1 m2 m3 r11 r12 r13 r21 r22 r23 r31 r32 r33;
% r1 = [r11; r12; r13; 1];
% r2 = [r21; r22; r23; 1];
% r3 = [r31; r32; r33; 1];
%%
% *******************************************************
% PRO CASTECNE OBECNE ODVOZENI MODELU - ZMENA VELIKOSTI A VAHY KULICKY
% *******************************************************
syms m3 bR;
% hmoty kazde vzajemne stacionarni casti modelu [kg]
m1 = 2.17500430; % Ramenol + kabel_osa2 ze SolidWorks
m2 = 2.00770144; % Rameno2 ze SolidWorks
% m3 = 7900*4*pi*bR^3/3; % hmotnost kulicky vypoctena ze zadaneho polomeru

% pozice teziste v lokalni souradne soustave kazde casti [m]
r1 = [-0.02983413; 0.01371001; -0.37684365; 1]; % v LSS_final
r2 = [-0.02081070; 0.00004702; -0.05392745; 1]; % v LSS_final
r3 = [0; 0; 0; 1];  % v LSS_final

% konstrukcni parametry [m] a gravitacni zrychleni [m*s^-2]
c = 0.3345;
g = 9.81;

%%
% *******************************************************
% PRIPRAVA NA AUTOMATIZOVANY VYPOCET
hmota = [m1, m2, m3, m3, 0, 0];   % hmota jednotlivych casti modelu
```

```
vec_r = sym(zeros(4, DOF));
vec_r(:,1) = r1;
vec_r(:,2) = r2;
vec_r(:,3) = r3;
vec_r(:,4) = r3;
vec_r(:,5) = r3;
vec_r(:,6) = r3;

%%
% TRANSFORMACNI MATICE MEZI SOURADNYMI SOUSTAVAMI
% ***************************************************
% Transformacni matice z lokalni do globalni souradne soustavy jednotlivych "clanku" spjatych s DOF
% cela transformace
T_ball = [  sin(fiY)*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + cos(fiY)*(cos(alfa)*sin(fiX)*sin(gama) -
cos(fiX)*cos(gama)*sin(beta) + cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(fiY)*(cos(beta)*cos(gama) +
sin(alfa)*sin(beta)*sin(gama)) - sin(fiY)*(cos(alfa)*sin(fiX)*sin(gama) - cos(fiX)*cos(gama)*sin(beta) +
cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(beta)*sin(alfa)*sin(fiX)*sin(gama) - cos(gama)*sin(beta)*sin(fiX) -
cos(alfa)*cos(fiX)*sin(gama),   Sx + x*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + c*sin(gama) -
bR*(cos(gama)*sin(beta) - cos(beta)*sin(alfa)*sin(gama)) - c*cos(alfa)*sin(gama) - y*cos(alfa)*sin(gama);
           cos(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) + cos(beta)*cos(fiX)*cos(gama)*sin(alfa))
- sin(fiY)*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)),   - cos(fiY)*(cos(beta)*sin(gama) -
cos(gama)*sin(alfa)*sin(beta)) - sin(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) +
cos(beta)*cos(fiX)*cos(gama)*sin(alfa)), sin(beta)*sin(fiX)*sin(gama) - cos(alfa)*cos(fiX)*cos(gama) +
cos(beta)*cos(gama)*sin(alfa)*sin(fiX),   c*cos(gama) - x*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)) - Sy +
bR*(sin(beta)*sin(gama) + cos(beta)*cos(gama)*sin(alfa)) - c*cos(alfa)*cos(gama) - y*cos(alfa)*cos(gama);
           cos(fiY)*sin(alfa)*sin(fiX) - cos(alfa)*sin(beta)*sin(fiY) - cos(alfa)*cos(beta)*cos(fiX)*cos(fiY),
cos(alfa)*cos(beta)*cos(fiX)*sin(fiY) - sin(alfa)*sin(fiX)*sin(fiY) - cos(alfa)*cos(fiY)*sin(beta),
- cos(fiX)*sin(alfa) - cos(alfa)*cos(beta)*sin(fiX),                                                    - c*sin(alfa) -
y*sin(alfa) - bR*cos(alfa)*cos(beta) - x*cos(alfa)*sin(beta);
           0,
0,
0,                                                                                                      1];

% alfa
T1 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, 0, 0, 0]);

% alfa + beta
T2 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, beta, 0, 0, 0, 0]);

% beta + x
T3 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[0, beta, x, 0, 0, 0]);

% alfa + y
T4 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, y, 0, 0]);

% alfa + fiX
T5 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, 0, fiX, 0]);

% beta + fiY
T6 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[0, beta, 0, 0, 0, fiY]);

T_mat = sym(zeros(4, 4, DOF));
T_mat(:,:,1) = T1;
T_mat(:,:,2) = T2;
T_mat(:,:,3) = T3;
T_mat(:,:,4) = T4;
T_mat(:,:,5) = T5;
T_mat(:,:,6) = T6;
%%
% MATICE SETRVACNOSTI V LOKALNI SOURADNE SOUSTAVE JEDNOTLIVYCH CLANKU
% ***************************************************
% jednotky jsou kilogramy * metry ctverecni
 % matice setrvacnosti 1. clanku, vzhledem ke KONCOVEMU LSS
I1 = [  0.31564756,     -0.00605803,    0.02443856;
        -0.00605803,    0.32040654,     -0.01076634;
        0.02443856,     -0.01076634,    0.01574458];

% matice setrvacnosti 2. clanku, vzhledem ke vzhledem ke KONCOVEMU LSS
I2 = [  0.12812298,     -0.00002008,    0.00255329;
        -0.00002008,    0.07986461,     0.00021746;
        0.00255329,     0.00021746,     0.05051258];

% matice setrvacnosti kulicky, vzhledem k jejimu lokalnimu souradnemu systemu
% obecne zadani pro homogenni kouli
I3 = [  2/5*m3*bR^2,    0,              0;
        0,              2/5*m3*bR^2,    0;
        0,              0,              2/5*m3*bR^2];

% kvuli zjednoduseni na hmotny bod musim zavest specialni promennou "typ", jejiz vyznam je uveden nize. Zde "typ" = 1
typ = 1;
%%
% PRO ZJEDNODUSENI NA HMOTNY BOD BUDOU MATICE SETRVACNOSTI NULOVE
% ***************************************************
% jednotky jsou kilogramy * metry ctverecni
% matice setrvacnosti 1. clanku, vzhledem k jeho lokalnimu souradnemu systemu
```

```
% I1 = zeros(3,3);
%
% % matice setrvacnosti 2. clanku, vzhledem k jeho lokalnimu souradnemu systemu
% I2 = zeros(3,3);
%
% % matice setrvacnosti kulicky, vzhledem k jejimu lokalnimu souradnemu systemu
% I3 = zeros(3,3);
%
% % pro pouziti zjednoduseni na hmotne body je potreba jeste v matici pseudosetrvacnosti vynulovat posledni radek, jinak
% % se pronasobi posledni radek s vektorem translace v transformacni matici a pruser je na svete. Zkratka definice matice
% % pseudosetrvacnosti je napasovana na matici setrvacnosti, ne na hmotny bod, proto je nutna tato uprava. Promenna "typ" =
0
% typ = 0;
%%
I = sym(zeros(3, 3, DOF));
I(:,:,1) = I1;
I(:,:,2) = I2;
I(:,:,3) = zeros(3, 3);
I(:,:,4) = zeros(3, 3);
I(:,:,5) = I3;
I(:,:,6) = I3;

% MATICE PSEUDOSETRVACNOSTI
% *****************************************************
Ip = sym(zeros(4, 4, DOF));
for i = 1:DOF
    Ip(:,:,i) = [(-I(1,1,i)+I(2,2,i)+I(3,3,i))/2,           I(1,2,i),           I(1,3,i),           hmota(i)*vec_r(1,i);...
        I(2,1,i),             (I(1,1,i)-I(2,2,i)+I(3,3,i))/2,         I(2,3,i),         hmota(i)*vec_r(2,i);...
        I(3,1,i),           I(3,2,i),                   (I(1,1,i)+I(2,2,i)-I(3,3,i))/2,     hmota(i)*vec_r(3,i);...
        hmota(i)*vec_r(1,i)*typ,        hmota(i)*vec_r(2,i)*typ,        hmota(i)*vec_r(3,i)*typ,        hmota(i)];
end

% URCENI MATIC DO POHYBOVYCH ROVNIC
% *****************************************************
par(1) = sym('alfa');
par(2) = sym('beta');
par(3) = sym('x');
par(4) = sym('y');
par(5) = sym('fiX');
par(6) = sym('fiY');

par_1d(1) = sym('alfa_1d');
par_1d(2) = sym('beta_1d');
par_1d(3) = sym('x_1d');
par_1d(4) = sym('y_1d');
par_1d(5) = sym('fiX_1d');
par_1d(6) = sym('fiY_1d');

par_2d(1,1) = sym('alfa_2d');
par_2d(2,1) = sym('beta_2d');
par_2d(3,1) = sym('x_2d');
par_2d(4,1) = sym('y_2d');
par_2d(5,1) = sym('fiX_2d');
par_2d(6,1) = sym('fiY_2d');

D = sym(zeros(DOF, DOF));
H_pom = sym(zeros(DOF, DOF, DOF));
H = sym(zeros(DOF, 1));
G = sym(zeros(DOF, 1));

% automatizovany vypocet
% matice D
for i = 1:DOF
    for j = 1:DOF
        D(i,j) = 0;
        for r = max(i,j):DOF
            D(i,j) = D(i,j) + trace(diff(T_mat(:,:,r),par(j))*Ip(:,:,r)*transpose(diff(T_mat(:,:,r),par(i))));
        end
        D(i,j) = simplify(D(i,j));
    end
end

% matice H_pom - pomocna matice pro urceni finalni matice H - pro rehlednost, prepisuju obecny vzorec z maticove formy
for i = 1:DOF
    for k = 1:DOF
        for m = 1:DOF
            H_pom(i,k,m) = 0;
            for r = max([i,k,m]):DOF
                H_pom(i,k,m) = H_pom(i,k,m) +
trace(diff(diff(T_mat(:,:,r),par(k)),par(m))*Ip(:,:,r)*transpose(diff(T_mat(:,:,r),par(i))));
            end
            H_pom(i,k,m) = simplify(H_pom(i,k,m));
        end
    end
end
```

```
% matice H - finalni vypocet z pomocne matice H_pom vypoctene vyse
for i = 1:DOF
    H(i) = 0;
    for k = 1:DOF
        for m = 1:DOF
            H(i) = H(i) + H_pom(i,k,m)*par_1d(k)*par_1d(m);
        end
    end
    H(i) = simplify(H(i));
end

% matice G
for i = 1:DOF
    G(i) = 0;
    for r = i:DOF
        G(i) = G(i) - hmota(r)*[0, 0, g, 1]*diff(T_mat(:,:,r),par(i))*vec_r(:,r);
    end
    G(i) = simplify(G(i));
end

% *******************************************************
%              VYSLEDNE POHYBOVE ROVNICE
% *******************************************************
D = vpa(D,4);
H = vpa(H,4);
G = vpa(G,4);
% *******************************************************
Q = vpa(simplify(D*par_2d + H + G), 4);
Q = subs(Q,[fiX, fiY, fiX_1d, fiY_1d, fiX_2d, fiY_2d],[-y/bR, x/bR, -y_1d/bR, x_1d/bR, -y_2d/bR, x_2d/bR]);
Q = vpa(expand(simplify(Q)), 4);
%%
% pokusne vycisleni v ustalenem stavu
% alfa = 0;
% beta = 0;
% x = 0;
% y = 0;
% fiX = 0;
% fiY = 0;
%
% alfa_1d = 0;
% beta_1d = 0;
% x_1d = 0;
% y_1d = 0;
% fiX_1d = 0;
% fiY_1d = 0;
%
% alfa_2d = 0;
% beta_2d = 0;
% x_2d = 0;
% y_2d = 0;
% fiX_2d = 0;
% fiY_2d = 0;
%
% vpa(eval(Q),4)
%%
Q_orig = [...
    0.2716*alfa_2d + 4.623*cos(alfa) + 0.6366*sin(alfa) - 0.0009261*sin(alfa)*sin(beta) - 0.1716*alfa_2d*bR - ...
    0.02413*alfa_2d*cos(2.0*beta) + 0.1119*alfa_2d*m3 - 2.008e-5*alfa_2d*sin(2.0*beta) + 3.179*alfa_2d*bR^2 + ...
    0.000249*beta_2d*cos(beta) + 3.281*m3*cos(alfa) - 21.34*bR*sin(alfa) + 0.01142*beta_2d*sin(beta) + ...
    0.01142*beta_1d^2*cos(beta) - 0.000249*beta_1d^2*sin(beta) + 0.4099*cos(beta)*sin(alfa) - 9.81*bR*m3*sin(alfa) + ...
    alfa_2d*m3*y^2 + 9.81*m3*y*cos(alfa) - 0.5633*bR*beta_1d^2*cos(beta) - 19.7*bR*cos(beta)*sin(alfa) - ...
    0.04178*alfa_2d*bR*cos(2.0*beta) - 4.016e-5*alfa_1d*beta_1d*cos(2.0*beta) + 9.44e-5*alfa_2d*bR*sin(2.0*beta) + ...
    0.04826*alfa_1d*beta_1d*sin(2.0*beta) + 0.669*alfa_2d*m3*y + 0.669*alfa_1d*m3*y_1d - 1.4*bR*m3*y_2d + ...
    1.004*alfa_2d*bR^2*cos(2.0*beta) + 1.4*alfa_2d*bR^2*m3 - 0.5633*bR*beta_2d*sin(beta) + 2.0*alfa_1d*m3*y*y_1d - ...
    2.008*alfa_1d*bR^2*beta_1d*sin(2.0*beta) + 0.0001888*alfa_1d*bR*beta_1d*cos(2.0*beta) +
0.08356*alfa_1d*bR*beta_1d*sin(2.0*beta);

    0.05051*beta_2d - 0.08356*bR*beta_2d + 2.008*bR^2*beta_2d + 2.008e-5*alfa_1d^2*cos(2.0*beta) +
0.000249*alfa_2d*cos(beta) - ...
    0.02413*alfa_1d^2*sin(2.0*beta) + 0.01142*alfa_2d*sin(beta) + 0.0009261*cos(alfa)*cos(beta) + 0.4099*cos(alfa)*sin(beta)
- ...
    9.81*bR*m3*sin(beta) + beta_2d*m3*x^2 + 9.81*m3*x*cos(beta) + 1.004*alfa_1d^2*bR^2*sin(2.0*beta) - ...
    19.7*bR*cos(alfa)*sin(beta) - 1.4*bR*m3*x_2d - 9.44e-5*alfa_1d^2*bR*cos(2.0*beta) + 1.4*bR^2*beta_2d*m3 - ...
    0.04178*alfa_1d^2*bR*sin(2.0*beta) - 0.5633*alfa_2d*bR*sin(beta) + 2.0*beta_1d*m3*x*x_1d;

    - 1.0*m3*x*beta_1d^2 + 1.0*m3*x_2d + 9.81*m3*sin(beta) - 1.0*bR*beta_2d*m3;

    m3*y_2d - 0.3345*alfa_1d^2*m3 + 9.81*m3*sin(alfa) - 1.0*alfa_1d^2*m3*y - 1.0*alfa_2d*bR*m3;

    0.4*alfa_2d*m3*bR^2 - 0.4*m3*y_2d*bR;

    - 0.4*beta_2d*m3*bR^2 + 0.4*m3*x_2d*bR;
   ...
   ];
```

```
Q_orig = vpa(Q_orig, 4);
%%
% ZJEDNODUSENY MATEMATICKY MODEL
% ******************************************************
% zjednodusuju tak, ze vyrazuju jednotlive cleny rovnice podle jejich ocekavanemu vlivu na staticke i dynamicke vlastnosti
sestavy. Jinymi slovy postupuju tak, ze ponechavam cleny s vyznamnymi koeficienty/hodnotami, a ty, ktere bych vyradil, tak do
jejich neznamych (derivace a proste vsechny nezname) dosadim pracovni bod (obecne nulove uhly a polohy kulicky, nulove
derivace, a za polomer i hmotnost kulicky dosadim prumer, s kterym budu pracovat - m3 = 0.5 kg a bR = 0.025 m - protoze to je
proste lepsi, nez bych je uplne vyhodil). Vysledek pak prictu ke clenu s nejvyznamnejsim koeficientem. Kazdou upravu, u ktere
tusim, ze by mohla zmenit dynamiku, otestuju v grafickem porovnani originalniho matematickeho modelu a toho zjednoduseneho na
typovem vstupnim prubehu. Prubeh musi byt co nejdynamictejsi, ktery budu po systemu pozadovat, protoze cim je vyssi dynamika,
tim vetsi maji vliv cleny, ktere se chystam z rovnice vypustit (protoze obecne vyrazuju souciny a mocniny derivaci, ktere
jsou vynasobene malymi cisly). Tento postup nejde jednoduse algoritmizovat, protoze bych musel celou symnobickou rovnici
rozparsovat a pak zvolit nejakou strategii, jak bych zjistoval, co je vyznamne a co ne a paralelne bych vsechno kontroloval
na typovem dynamickem prubehu...a to by byl proste mazec napsat!
% ZJISTENO, ZE DOSAZOVANI PRACOVNIHO BODU VYRAZENYCH CLENU A PRICTENI JEJICH VLIVU DO POHYBOVE ROVNICE VEDE VETSINOU JEN K
MALEMU ZLEPSENI PRESNOSTI (v radech tisicin az desetitisicin), PROTOZE VYRAZENE CLENY BYLY OPRAVDU JEN MALO VYZNAMNE. TENTO
PRISTUP BYCH POUZIL JEN NA MODEL, U KTEREHO BY SE VYZNAMNE vs NEVYZNAMNE CLENY LISILY JEN MALO. TADY TO V PODSTATE NEMA SMYSL,
STACI JEN MAZAT A KONTROLOVAT DYNAMICKE PRUBEHY. AZ SE ZACNOU LISIT (musi odhadnout uzivatel co je moc a co malo, zkratka
zvolit kompromis), TAK S MAZANIM PRESTANU A ROVNICI POVAZUJU ZA FINALNI ZJEDNODUSENOU
Q_zjedn = [...
    0.2716*alfa_2d + (4.623 + 3.281*m3 + 9.81*m3*y)*cos(alfa);

    0.05051*beta_2d + 9.81*m3*x*cos(beta);

    m3*x_2d + 9.81*m3*sin(beta) - bR*beta_2d*m3;

    m3*y_2d - 0.3345*alfa_1d^2*m3 + 9.81*m3*sin(alfa) - alfa_2d*bR*m3;

    - 0.4*m3*y_2d*bR + 0.4*alfa_2d*m3*bR^2;

    0.4*m3*x_2d*bR - 0.4*beta_2d*m3*bR^2;
   ...
   ];

%%
% prepis zpet do matic D, H a G pro zjednodusenou formu pohybovych rovnic - varianta pro 6 DOF
% vektor zrychleni kterým se násobí je v tomto tvaru: [alfa_2d; beta_2d; x_2d; y_2d; y_2d; x_2d]
D_zjedn =   [   0.2716,         0,              0,      0,      0,          0;
                0,              0.05051,        0,      0,      0,          0;
                0,              -bR*m3,         m3,     0,      0,          0;
                -bR*m3,         0,              0,      m3,     0,          0;
                0.4*m3*bR^2,    0,              0,      0,      -0.4*m3*bR,  0;
                0,              -0.4*m3*bR^2,   0,      0,      0,          0.4*m3*bR];

H_zjedn =   [   0;
                0;
                0;
                - 0.3345*alfa_1d^2*m3;
                0;
                0];

G_zjedn =   [   4.623*cos(alfa) + 3.281*m3*cos(alfa) + 9.81*m3*y*cos(alfa);
                9.81*m3*x*cos(beta);
                9.81*m3*sin(beta);
                9.81*m3*sin(alfa);
                0;
                0];

%%
% kontrola spravnosti prepsani pohybovych rovnic do matic D_zjedn, H_zjedn a G_zjedn
if isequal(vpa(expand(D_zjedn*[alfa_2d; beta_2d; x_2d; y_2d; y_2d; x_2d] + H_zjedn + G_zjedn),4),vpa(expand(Q_zjedn),4))
    disp('Prepsano spravne');
else
    disp('Prepsano CHYBNE!!!');
end

%%
% PRIPRAVA NA ZAPIS DO STAVOVEHO PROSTORU
% ******************************************************
% Po prepsani do tvaru L*[alfa_2d; beta_2d; x_2d; y_2d; y_2d; x_2d] = K lze psat
% ******************************************************
syms Q_alfa Q_beta Q_x Q_y;
L = D_zjedn;
K = [Q_alfa; Q_beta; 0; 0; 0; 0] - H_zjedn - G_zjedn;

vysl = inv(L)*K;

alfa_2d = vpa(simplify(vysl(1),'Steps',100),4);
beta_2d = vpa(simplify(vysl(2),'Steps',100),4);
x_2d = vpa(simplify(vysl(3),'Steps',100),4);
y_2d = vpa(simplify(vysl(4),'Steps',100),4);
```

PŘÍLOHA C: MATEMATICKÝ MODEL - MATLAB SKRIPT - POSTUP 2 - 4 DOF

```
% close all;  clear all;  clc;

% symbolicke promenne pouzite v tranformacich a pri derivacich (derivuju jen stavove promenne, takze 1. a 2. derivace DOF)
syms Sx Sy x y bR alfa beta gama c g fiX fiY par par_1d par_2d;
syms alfa_1d alfa_2d beta_1d beta_2d x_1d x_2d y_1d y_2d;

% z pohledu pohybovych rovnic nezalezi na tom, jak bude natocena kamera vuci naklonen rovice (nezalezi na uhlu gama)...ten
zbytecne zeslozituje cely vyraz a podle me by mel spravne z vyslednych rovnic vypadnout, stejne jako napr. konstanty Sx a Sy.
Protoze je ale asi rotace posunuta a pak je goniometricka funkces argumentem "gama" vsim moznym pronasobena, tak uz se pak neda
vykratit a zjednodusit a napr. figuruje uz v druhe absolutni rychlosti. No a v absolutni rychlosti prece nemuze zalezet na tom,
kde a jak si umistim globalni souradnou soustavu. Zakladni myslenkou je, ze pohybove rovnice musi vypadat stejne bez ohledu na
to, kde se rozhodnu umistit globalni souradnou soustavu a proto si ji umistim ,,vhodneji" vynulovanim "gamy". To nemuze mit
zadny vliv na dynamiku pohybu naklonen roviny, kterou pohybova rovnice popisuej, protoze je vyloucene, abych natocenim kamery
zmenil dynamiku modelu. U kinematickych prepoctu u zpracovani obrazu pochopitelne byt musi, ale pro pohybove rovnice muzu
udelat kvalifikovane zjednoduseni. Po vycisleni a dosazeni konkretnich cisel pri reseni by nemel byt rozdil ve vysledku, ale
pro obecne vyjadreni a dalsi pocitani napr. v realnem case je zbytecne, abych mel slozitou rovnici, ktera vyjde stejne, jako
zjednodusena.
% OVERENO, ZE VYSE UVEDENE PLATI - PRI VOLBE JAKEKOLI "GAMY" JE VYSLEDNY PRUBEH DYNAMICKEHO CHOVANI STEJNY...TAKZE MUZU
VYNULOVAT
gama = 0;

% pro automaticke vyhodnoceni rozmeru matic D, H, G, poctu pohybovych rovnic apod.
DOF = 4;

%%
% HODNOTY ZE SolidWorks
% *******************************************************
% PRO ODVOZENI MODELU S KONKRETNIMI CISLY
% *******************************************************
% hmoty kazde vzajemne stacionarni casti modelu [kg]
% m1 = 2.17500430; % Ramenol + kabel_osa2 ze SolidWorks
% m2 = 2.00770144; % Rameno2 ze SolidWorks
% m3 = 0.26473154; % BalancniRovina_kulicka ze SolidWorks
%
% % pozice teziste v lokalni souradne soustave kazde casti [m]
% r1 = [-0.02983413; 0.01371001; -0.37684365; 1];   % v LSS_final
% r2 = [-0.02081070; 0.00004702; -0.05392745; 1]; % v LSS_final
% r3 = [0; 0; 0; 1];  % v LSS_final
%
% % konstrukcni parametry [m]
% c = 0.3345;
% bR = 0.02;
% g = 9.81;
%%
% *******************************************************
% PRO UPLNE OBECNE ODVOZENI MODELU
% *******************************************************
% syms g c bR m1 m2 m3 r11 r12 r13 r21 r22 r23 r31 r32 r33;
% r1 = [r11; r12; r13; 1];
% r2 = [r21; r22; r23; 1];
% r3 = [r31; r32; r33; 1];
%%
% *******************************************************
% PRO CASTECNE OBECNE ODVOZENI MODELU - ZMENA VELIKOSTI A VAHY KULICKY
% *******************************************************
syms m3 bR g;
% hmoty kazde vzajemne stacionarni casti modelu [kg]
m1 = 2.17500430; % Ramenol + kabel_osa2 ze SolidWorks
m2 = 2.00770144; % Rameno2 ze SolidWorks
% m3 = 7900*4*pi*bR^3/3; % hmotnost kulicky vypoctena ze zadaneho polomeru

% pozice teziste v lokalni souradne soustave kazde casti [m]
r1 = [-0.02983413; 0.01371001; -0.37684365; 1]; % v LSS_final
r2 = [-0.02081070; 0.00004702; -0.05392745; 1]; % v LSS_final
r3 = [0; 0; 0; 1];  % v LSS_final

% konstrukcni parametry [m] a gravitacni zrychleni [m*s^-2]
c = 0.3345;
```

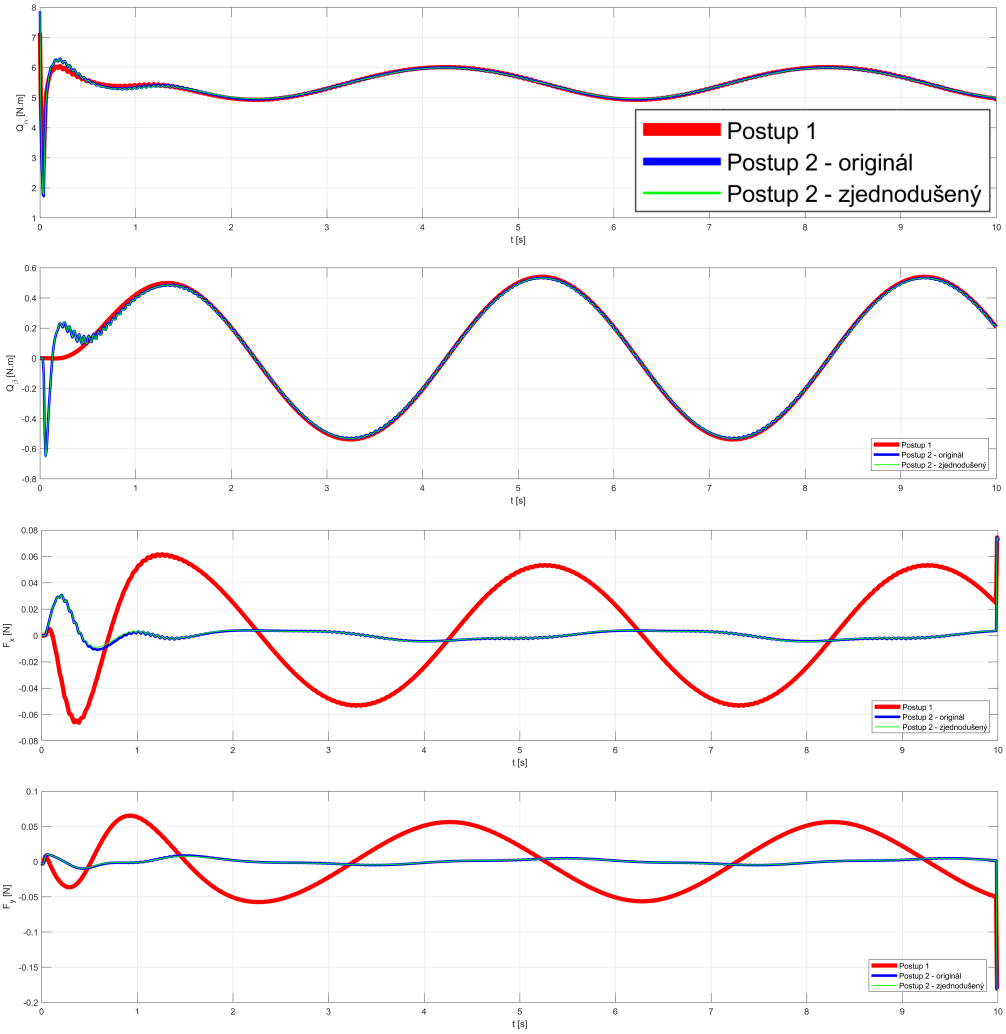
```
%%
% ******************************************************
% PRIPRAVA NA AUTOMATIZOVANY VYPOCET
hmota = [m1, m2, m3, m3];   % hmota jednotlivych casti modelu

vec_r = sym(zeros(4, DOF));
vec_r(:,1) = r1;
vec_r(:,2) = r2;
vec_r(:,3) = r3;
vec_r(:,4) = r3;

%%
% TRANSFORMACNI MATICE MEZI SOURADNYMI SOUSTAVAMI
% ******************************************************
% Transformacni matice z lokalni do globalni souradne soustavy jednotlivych "clanku" spjatych s DOF
% cela transformace
T_ball = [  sin(fiY)*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + cos(fiY)*(cos(alfa)*sin(fiX)*sin(gama) -
cos(fiX)*cos(gama)*sin(beta) + cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(fiY)*(cos(beta)*cos(gama) +
sin(alfa)*sin(beta)*sin(gama)) - sin(fiY)*(cos(alfa)*sin(fiX)*sin(gama) - cos(fiX)*cos(gama)*sin(beta) +
cos(beta)*cos(fiX)*sin(alfa)*sin(gama)),   cos(beta)*sin(alfa)*sin(fiX)*sin(gama) - cos(gama)*sin(beta)*sin(fiX) -
cos(alfa)*cos(fiX)*sin(gama),   Sx + x*(cos(beta)*cos(gama) + sin(alfa)*sin(beta)*sin(gama)) + c*sin(gama) -
bR*(cos(gama)*sin(beta) - cos(beta)*sin(alfa)*sin(gama)) - c*cos(alfa)*sin(gama) - y*cos(alfa)*sin(gama);
           cos(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) + cos(beta)*cos(fiX)*cos(gama)*sin(alfa)) -
sin(fiY)*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)),   - cos(fiY)*(cos(beta)*sin(gama) -
cos(gama)*sin(alfa)*sin(beta)) - sin(fiY)*(cos(alfa)*cos(gama)*sin(fiX) + cos(fiX)*sin(beta)*sin(gama) +
cos(beta)*cos(fiX)*cos(gama)*sin(alfa)), sin(beta)*sin(fiX)*sin(gama) - cos(alfa)*cos(fiX)*cos(gama) +
cos(beta)*cos(gama)*sin(alfa)*sin(fiX),   c*cos(gama) - x*(cos(beta)*sin(gama) - cos(gama)*sin(alfa)*sin(beta)) - Sy +
bR*(sin(beta)*sin(gama) + cos(beta)*cos(gama)*sin(alfa)) - c*cos(alfa)*cos(gama) - y*cos(alfa)*cos(gama);
           cos(fiY)*sin(alfa)*sin(fiX) - cos(alfa)*sin(beta)*sin(fiY) - cos(alfa)*cos(beta)*cos(fiX)*cos(fiY),
cos(alfa)*cos(beta)*cos(fiX)*sin(fiY) - sin(alfa)*sin(fiX)*sin(fiY) - cos(alfa)*cos(fiY)*sin(beta),
- cos(fiX)*sin(alfa) - cos(alfa)*cos(beta)*sin(fiX),                                                    - c*sin(alfa) -
y*sin(alfa) - bR*cos(alfa)*cos(beta) - x*cos(alfa)*sin(beta);
           0,
0,
0,                                                                                                     1];

% alfa
T1 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, 0, 0, 0]);

% alfa + beta
T2 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, beta, 0, 0, 0, 0]);

% beta + x
T3 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[0, beta, x, 0, 0, x/bR]);

% alfa + y
T4 = subs(T_ball,[alfa, beta, x, y, fiX, fiY],[alfa, 0, 0, y, -y/bR, 0]);

T_mat = sym(zeros(4, 4, DOF));
T_mat(:,:,1) = T1;
T_mat(:,:,2) = T2;
T_mat(:,:,3) = T3;
T_mat(:,:,4) = T4;
%%
% MATICE SETRVACNOSTI V LOKALNI SOURADNE SOUSTAVE JEDNOTLIVYCH CLANKU
% ******************************************************
% jednotky jsou kilogramy * metry ctverecni
 % matice setrvacnosti 1. clanku, vzhledem ke KONCOVEMU LSS
I1 = [  0.31564756,     -0.00605803,    0.02443856;
        -0.00605803,    0.32040654,     -0.01076634;
        0.02443856,     -0.01076634,    0.01574458];

% matice setrvacnosti 2. clanku, vzhledem ke vzhledem ke KONCOVEMU LSS
I2 = [  0.12812298,     -0.00002008,    0.00255329;
        -0.00002008,    0.07986461,     0.00021746;
        0.00255329,     0.00021746,     0.05051258];

% matice setrvacnosti kulicky, vzhledem k jejimu lokalnimu souradnemu systemu
% obecne zadani pro homogenni kouli
I3 = [  2/5*m3*bR^2,    0,              0;
        0,              2/5*m3*bR^2,    0;
        0,              0,              2/5*m3*bR^2];
```

```
% kvuli zjednoduseni na hmotny bod musim zavest specialni promennou "typ", jejiz vyznam je uveden nize. Zde "typ" = 1
typ = 1;
%%
% PRO ZJEDNODUSENI NA HMOTNY BOD BUDOU MATICE SETRVACNOSTI NULOVE
% ******************************************************
% jednotky jsou kilogramy * metry ctverecni
% matice setrvacnosti 1. clanku, vzhledem k jeho lokalnimu souradnemu systemu
% I1 = zeros(3,3);
%
% % matice setrvacnosti 2. clanku, vzhledem k jeho lokalnimu souradnemu systemu
% I2 = zeros(3,3);
%
% % matice setrvacnosti kulicky, vzhledem k jejimu lokalnimu souradnemu systemu
% I3 = zeros(3,3);
%
% % pro pouziti zjednoduseni na hmotne body je potreba jeste v matici pseudosetrvacnosti vynulovat posledni radek, jinak
% % se pronasobi posledni radek s vektorem translace v transformacni matici a pruser je na svete. Zkratka definice matice
% % pseudosetrvacnosti je napasovana na matici setrvacnosti, ne na hmotny bod, proto je nutna tato uprava. Promenna "typ" = 0
% typ = 0;
%%
I = sym(zeros(3, 3, DOF));
I(:,:,1) = I1;
I(:,:,2) = I2;
I(:,:,3) = I3;
I(:,:,4) = I3;

% MATICE PSEUDOSETRVACNOSTI
% ******************************************************
Ip = sym(zeros(4, 4, DOF));
for i = 1:DOF
    Ip(:,:,i) = [(-I(1,1,i)+I(2,2,i)+I(3,3,i))/2,           I(1,2,i),           I(1,3,i),           hmota(i)*vec_r(1,i);...
        I(2,1,i),               (I(1,1,i)-I(2,2,i)+I(3,3,i))/2,         I(2,3,i),           hmota(i)*vec_r(2,i);...
        I(3,1,i),           I(3,2,i),                   (I(1,1,i)+I(2,2,i)-I(3,3,i))/2,     hmota(i)*vec_r(3,i);...
        hmota(i)*vec_r(1,i)*typ,        hmota(i)*vec_r(2,i)*typ,        hmota(i)*vec_r(3,i)*typ,        hmota(i)];
end

% URCENI MATIC DO POHYBOVYCH ROVNIC
% ******************************************************
par(1) = sym('alfa');
par(2) = sym('beta');
par(3) = sym('x');
par(4) = sym('y');

par_1d(1) = sym('alfa_1d');
par_1d(2) = sym('beta_1d');
par_1d(3) = sym('x_1d');
par_1d(4) = sym('y_1d');

par_2d(1,1) = sym('alfa_2d');
par_2d(2,1) = sym('beta_2d');
par_2d(3,1) = sym('x_2d');
par_2d(4,1) = sym('y_2d');

D = sym(zeros(DOF, DOF));
H_pom = sym(zeros(DOF, DOF, DOF));
H = sym(zeros(DOF, 1));
G = sym(zeros(DOF, 1));

% automatizovany vypocet
% matice D
for i = 1:DOF
    for j = 1:DOF
        D(i,j) = 0;
        for r = max(i,j):DOF
            D(i,j) = D(i,j) + trace(diff(T_mat(:,:,r),par(j))*Ip(:,:,r)*transpose(diff(T_mat(:,:,r),par(i))));
        end
        D(i,j) = simplify(D(i,j));
    end
end

% matice H_pom - pomocna matice pro urceni finalni matice H - pro rehlednost, prepisuju obecny vzorec z maticove formy
for i = 1:DOF
    for k = 1:DOF
        for m = 1:DOF
            H_pom(i,k,m) = 0;
            for r = max([i,k,m]):DOF
                H_pom(i,k,m) = H_pom(i,k,m) +
trace(diff(diff(T_mat(:,:,r),par(k)),par(m))*Ip(:,:,r)*transpose(diff(T_mat(:,:,r),par(i))));
            end
            H_pom(i,k,m) = simplify(H_pom(i,k,m));
        end
    end
end
```

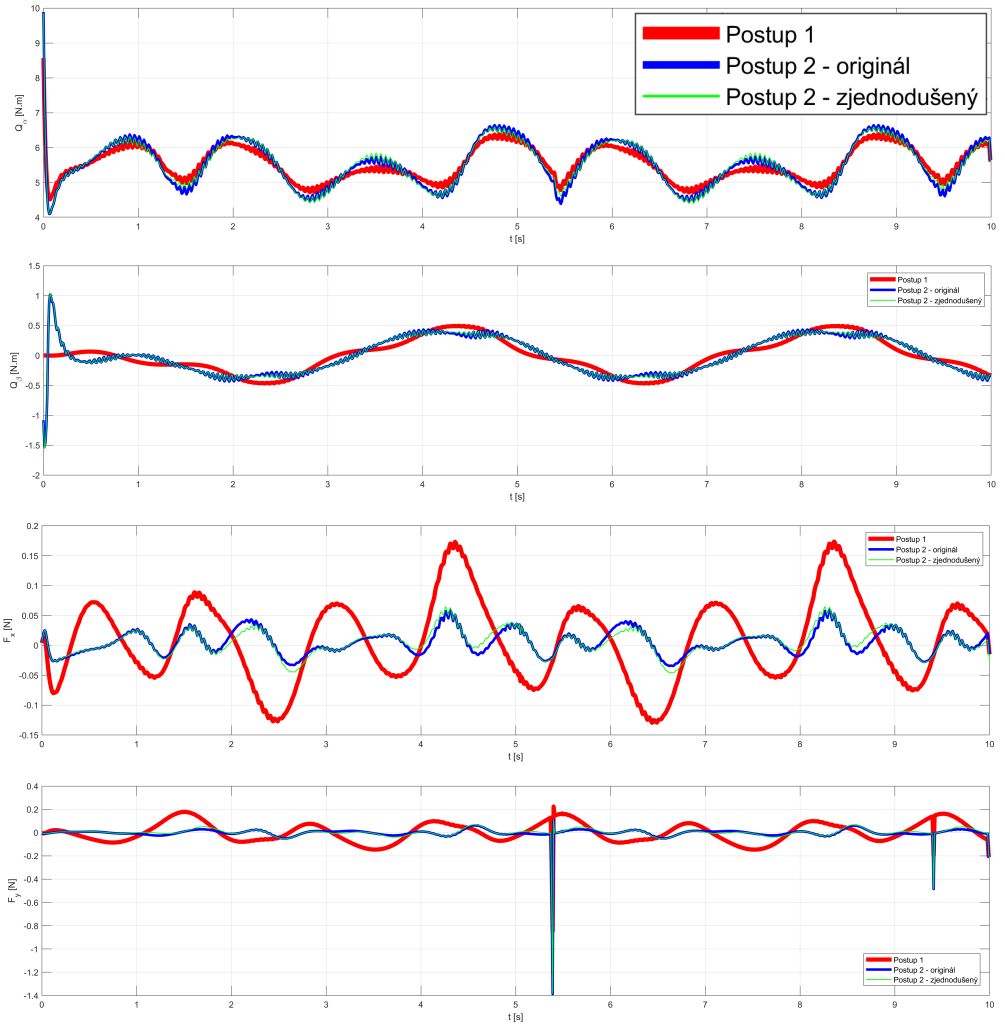
```
% matice H - finalni vypocet z pomocne matice H_pom vypoctene vyse
for i = 1:DOF
    H(i) = 0;
    for k = 1:DOF
        for m = 1:DOF
            H(i) = H(i) + H_pom(i,k,m)*par_1d(k)*par_1d(m);
        end
    end
    H(i) = simplify(H(i));
end

% matice G
for i = 1:DOF
    G(i) = 0;
    for r = i:DOF
        G(i) = G(i) - hmota(r)*[0, 0, g, 1]*diff(T_mat(:,:,r),par(i))*vec_r(:,r);
    end
    G(i) = simplify(G(i));
end

% *********************************************************
%              VYSLEDNE POHYBOVE ROVNICE
% *********************************************************
D = vpa(D,4);
H = vpa(H,4);
G = vpa(G,4);
% *********************************************************
Q = vpa(simplify(D*par_2d + H + G), 4);
Q = vpa(expand(simplify(Q)), 4);
%%
% pokusne vycisleni v ustalenem stavu
% alfa = 0;
% beta = 0;
% x = 0;
% y = 0;
%
% alfa_1d = 0;
% beta_1d = 0;
% x_1d = 0;
% y_1d = 0;
%
% alfa_2d = 0;
% beta_2d = 0;
% x_2d = 0;
% y_2d = 0;
%
% vpa(eval(Q),4)
%%
Q_orig = [...
     0.2716*alfa_2d - 0.1716*alfa_2d*bR - 0.02413*alfa_2d*cos(2.0*beta) + 0.1119*alfa_2d*m3 - 2.008e-5*alfa_2d*sin(2.0*beta)
+ ...
     3.179*alfa_2d*bR^2 + 0.000249*beta_2d*cos(beta) + 0.4712*g*cos(alfa) + 0.01142*beta_2d*sin(beta) + 0.06489*g*sin(alfa) +
...
     0.01142*beta_1d^2*cos(beta) - 0.000249*beta_1d^2*sin(beta) + alfa_2d*m3*y^2 - 0.5633*bR*beta_1d^2*cos(beta) + ...
     0.04178*g*cos(beta)*sin(alfa) - 9.44e-5*g*sin(alfa)*sin(beta) - 0.04178*alfa_2d*bR*cos(2.0*beta) - ...
     4.016e-5*alfa_1d*beta_1d*cos(2.0*beta) + 9.44e-5*alfa_2d*bR*sin(2.0*beta) + 0.04826*alfa_1d*beta_1d*sin(2.0*beta) + ...
     0.669*alfa_2d*m3*y + 0.669*alfa_1d*m3*y_1d - 1.4*bR*m3*y_2d + 1.004*alfa_2d*bR^2*cos(2.0*beta) + 1.4*alfa_2d*bR^2*m3 +
...
     0.3345*g*m3*cos(alfa) - 0.5633*bR*beta_2d*sin(beta) - 2.175*bR*g*sin(alfa) + 2.0*alfa_1d*m3*y*y_1d - ...
     2.008*alfa_1d*bR^2*beta_1d*sin(2.0*beta) - 1.0*bR*g*m3*sin(alfa) + 1.0*g*m3*y*cos(alfa) - 2.008*bR*g*cos(beta)*sin(alfa)
+ ...
     0.0001888*alfa_1d*bR*beta_1d*cos(2.0*beta) + 0.08356*alfa_1d*bR*beta_1d*sin(2.0*beta);

    0.05051*beta_2d - 0.08356*bR*beta_2d + 2.008*bR^2*beta_2d + 2.008e-5*alfa_1d^2*cos(2.0*beta) + 0.000249*alfa_2d*cos(beta)
- ...
    0.02413*alfa_1d^2*sin(2.0*beta) + 0.01142*alfa_2d*sin(beta) + beta_2d*m3*x^2 + 9.44e-5*g*cos(alfa)*cos(beta) + ...
    1.004*alfa_1d^2*bR^2*sin(2.0*beta) + 0.04178*g*cos(alfa)*sin(beta) - 1.4*bR*m3*x_2d - 9.44e-5*alfa_1d^2*bR*cos(2.0*beta)
+ ...
    1.4*bR^2*beta_2d*m3 - 0.04178*alfa_1d^2*bR*sin(2.0*beta) - 0.5633*alfa_2d*bR*sin(beta) + 2.0*beta_1d*m3*x*x_1d - ...
    1.0*bR*g*m3*sin(beta) + 1.0*g*m3*x*cos(beta) - 2.008*bR*g*cos(alfa)*sin(beta);

    - 1.0*m3*x*beta_1d^2 + 1.4*m3*x_2d + 1.0*g*m3*sin(beta) - 1.4*bR*beta_2d*m3;

    1.4*m3*y_2d - 0.3345*alfa_1d^2*m3 - 1.0*alfa_1d^2*m3*y + g*m3*sin(alfa) - 1.4*alfa_2d*bR*m3
   ...
   ];

Q_orig = vpa(Q_orig, 4);
```

```
%%
% ZJEDNODUSENY MATEMATICKY MODEL
% ******************************************************
% zjednodusuju tak, ze vyrazuju jednotlive cleny rovnice podle jejich ocekavanemu vlivu na staticke i dynamicke vlastnosti
sestavy. Jinymi slovy postupuju tak, ze ponechavam cleny s vyznamnymi koeficienty/hodnotami, a ty, ktere bych vyradil, tak do
jejich neznamych (derivace a proste vsechny nezname) dosadim pracovni bod (obecne nulove uhly a polohy kulicky, nulove derivace,
a za polomer i hmotnost kulicky dosadim prumer, s kterym budu pracovat - m3 = 0.5 kg a bR = 0.025 m - protoze to je proste
lepsi, nez bych je uplne vyhodil). Vysledek pak prictu ke clenu s nejvyznamnejsim koeficientem. Kazdou upravu, u ktere tusim,
ze by mohla zmenit dynamiku, otestuju v grafickem porovnani originalniho matematickeho modelu a toho zjednoduseneho na typovem
vstupnim prubehu. Prubeh musi byt co nejdynamictejsi, ktery budu po systemu pozadovat, protoze cim je vyssi dynamika, tim vetsi
maji vliv cleny, ktere se chystam z rovnice vypustit (protoze obecne vyrazuju souciny a mocniny derivaci, ktere jsou vynasobene
malymi cisly). Tento postup nejde jednoduse algoritmizovat, protoze bych musel celou symnobickou rovnici rozparsovat a pak
zvolit nejakou strategii, jak bych zjistoval, co je vyznamne a co ne a paralelne bych vsechno kontroloval na typovem dynamickem
prubehu...a to by byl proste mazec napsat!
% ZJISTENO, ZE DOSAZOVANI PRACOVNIHO BODU VYRAZENYCH CLENU A PRICTENI JEJICH VLIVU DO POHYBOVE ROVNICE VEDE VETSINOU JEN K
MALEMU ZLEPSENI PRESNOSTI (v radech tisicin az desetitisicin), PROTOZE VYRAZENE CLENY BYLY OPRAVDU JEN MALO VYZNAMNE. TENTO
PRISTUP BYCH POUZIL JEN NA MODEL, U KTEREHO BY SE VYZNAMNE vs NEVYZNAMNE CLENY LISILY JEN MALO. TADY TO V PODSTATE NEMA SMYSL,
STACI JEN MAZAT A KONTROLOVAT DYNAMICKE PRUBEHY. AZ SE ZACNOU LISIT (musi odhadnout uzivatel co je moc a co malo, zkratka
zvolit kompromis), TAK S MAZANIM PRESTANU A ROVNICI POVAZUJU ZA FINALNI ZJEDNODUSENOU
Q_zjedn = [...
    0.2716*alfa_2d + 0.4712*g*cos(alfa) + 0.3345*g*m3*cos(alfa)+ g*m3*y*cos(alfa);

    0.05051*beta_2d + g*m3*x*cos(beta);

    1.4*m3*x_2d + g*m3*sin(beta) - 1.4*bR*beta_2d*m3;

    1.4*m3*y_2d - 0.3345*m3*alfa_1d^2 + g*m3*sin(alfa) - 1.4*alfa_2d*bR*m3;
   ...
   ];

%%
% prepis zpet do matic D, H a G pro zjednodusenou formu pohybovych rovnic - varianta pro 4 DOF
% vektor zrychlení kterym se násobí je v tomto tvaru: [alfa_2d; beta_2d; x_2d; y_2d]
D_zjedn =   [   0.2716,         0,              0,              0;
                0,              0.05051,        0,              0;
                0,              -1.4*bR*m3,     1.4*m3,         0;
                -1.4*bR*m3,     0,                  0,          1.4*m3];

H_zjedn =   [   0;
                0;
                0;
                -0.3345*m3*alfa_1d^2];

G_zjedn =   [   0.4712*g*cos(alfa) + 0.3345*g*m3*cos(alfa)+ g*m3*y*cos(alfa);
                g*m3*x*cos(beta);
                g*m3*sin(beta);
                g*m3*sin(alfa)];
%%
% kontrola spravnosti prepsani pohybovych rovnic do matic D_zjedn, H_zjedn a G_zjedn
if isequal(vpa(expand(D_zjedn*[alfa_2d; beta_2d; x_2d; y_2d] + H_zjedn + G_zjedn),4),vpa(expand(Q_zjedn),4))
    disp('Prepsano spravne');
else
    disp('Prepsano CHYBNE!!!');
end

%%
% PRIPRAVA NA ZAPIS DO STAVOVEHO PROSTORU
% ******************************************************
% Po prepsani do tvaru L*[alfa_2d; beta_2d; x_2d; y_2d; y_2d; x_2d] = K lze psat
% ******************************************************
syms Q_alfa Q_beta Q_x Q_y;
L = D_zjedn;
K = [Q_alfa; Q_beta; 0; 0] - H_zjedn - G_zjedn;

vysl = inv(L)*K;

alfa_2d = vpa(simplify(vysl(1),'Steps',100),4);
beta_2d = vpa(simplify(vysl(2),'Steps',100),4);
x_2d = vpa(simplify(vysl(3),'Steps',100),4);
y_2d = vpa(simplify(vysl(4),'Steps',100),4);
```

PŘÍLOHA D: MATEMATICKÝ MODEL - TESTOVACÍ PRŮBĚH VE TVARU KRUŽNICE



PŘÍLOHA E: MATEMATICKÝ MODEL - TESTOVACÍ PRŮBĚH VE TVARU ASTEROIDY



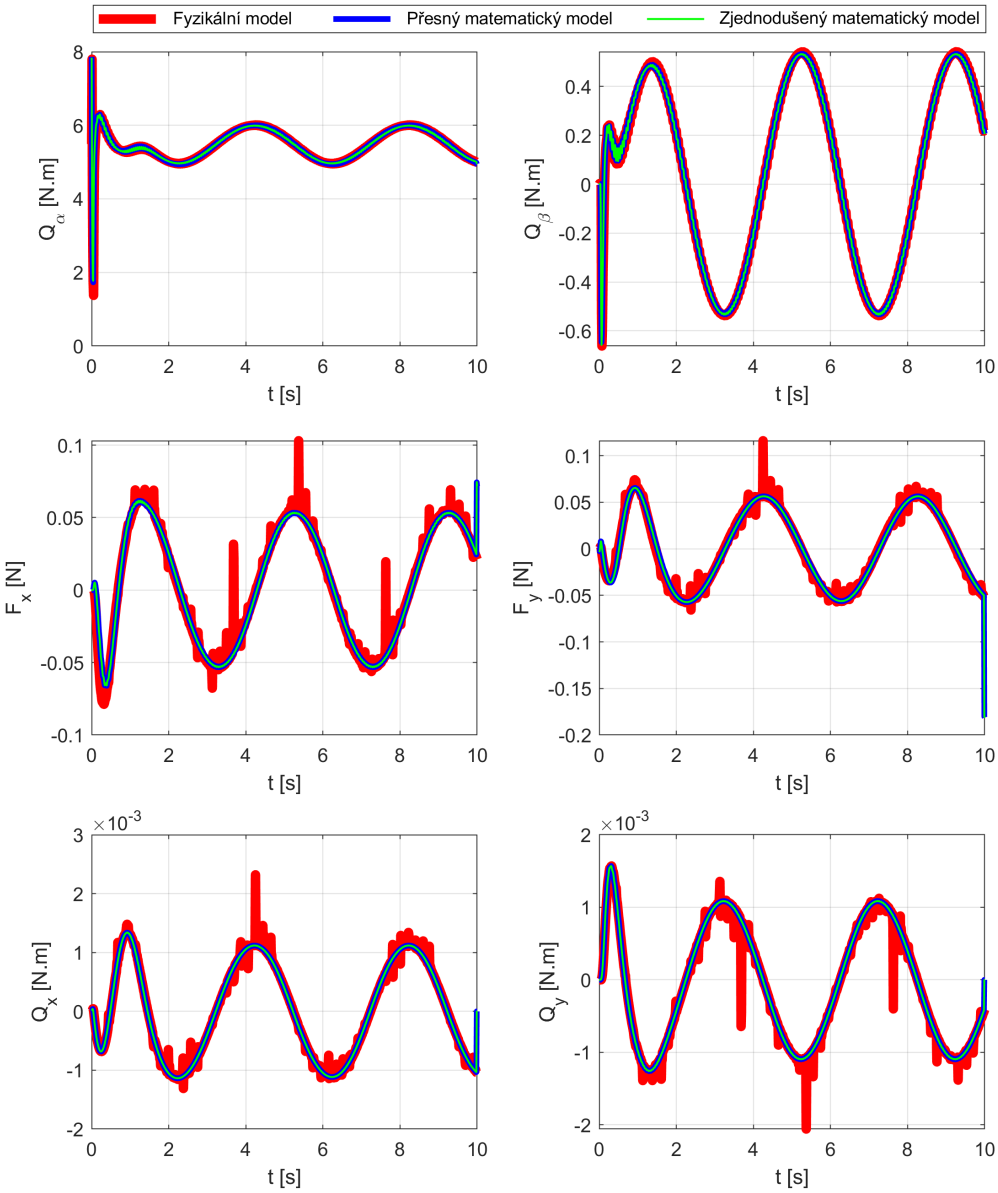
PŘÍLOHA F: FYZIKÁLNÍ MODEL - MATLAB SKRIPT PRO VAZBU KULIČKY S OKOLÍM

```
function [Fx, Fy, Fz, Qx, Qy, Qz, stop]  = fcn(px, vx, py, vy, pz, vz, wx, wy, wz, par)
% Construction parameters
% ***************************************
bR = par(1);          % Ball's radius [m]
platex = par(2);      % Plate width in X direction [m]
platey = par(3);      % Plate width in Y direction [m]
wallthick = par(4); % Wall thickness [m]
wallheight = par(5);% Wall height above the plate [m]

% Time threshold of friction characteristic
% ***************************************
% this parameter should be the same or similar at least for every material (?)
frtrh = par(6);      % Static friction threshold speed [m/s]" - the speed at which static friction is not rise, but it fall
down to kinematic friction value

% Plate's properties
% ***************************************
Kpen = par(7);        % Spring stiffness of plate/ball transition - hard stop spring constant [N/m]
Dpen = par(8);        % Damping coefficient of plate/ball transition - hard stop damping constant [N/(m/s)]

frstat = par(9);      % Static friction constant [-] - from the speed of 0 to "frtrh" it is increased by a straight line
frkin = par(10);      % Kinematic friction constant [-] - friction will decrease to this level after static friction peak
flin = par(11);       % Linear friction constant - in this case rolling resistance

% Wall's properties
% ***************************************
Kpenw = par(12)*par(17);     % Spring stiffness of wall/ball transition - hard stop spring constant [N/m]
Dpenw = par(13)*par(17);     % Damping coefficient of wall/ball transition - hard stop damping constant [N/(m/s)]

frstatw = par(14)*par(17);  % Static friction constant [-] - from the speed of 0 to "frtrh" it is increased by a straight
line
frkinw = par(15)*par(17);    % Kinematic friction constant [-] - friction will decrease to this level after static friction
peak
flinw = par(16)*par(17);     % Linear friction constant [-] - in this case rolling resistance
crushstr = par(18)*par(17); % Crushing strength of the material [N/m^2]
touchar = par(19)*par(17);  % Touching area between the ball and the wall [m^2]

% Circumferential speed - obvodová rychlost
vxc = bR*wy;     % Rotation in "x" direction is around "y" axis
vyc = -bR*wx;    % the same as previous, minus because of orientation of coordinate system, now + around X rotation is + in Y
translation also
vzc = bR*wz;

% Sliding speed - rychlost prokluzu
vxlin = vxc-vx;
vylin = vyc-vy;

% Inicialization
breaked = 0;

% Support force from plate - if the ball penetrate the plate, the plate acts on the ball with the opposite force, otherwise
the force is zero
if (pz < 0)
    Fz = -Kpen*pz - Dpen*vz;
else
    Fz = 0;
end

% if the ball is on the plate area, it rols. This rolling depends on static, kinematic and linear friction
if (abs(px)<(platex/2-bR-wallthick) && abs(py)<(platey/2-bR-wallthick))

    % Frictional force acting against the direction of movement
    Fx = friction(vxlin, vx, frstat, frkin, frtrh, flin)*Fz;
    Fy = friction(vylin, vy, frstat, frkin, frtrh, flin)*Fz;

    % Rolling torques - directions are against the directions of friction forces

    %
**************!!!!!!!!!!!!!!!!!!*******************!!!!!!!!!!!!!!!!!!*******************!!!!!!!!!!!!!!!!!!*******************
    % FROM TORQUES HAS TO BE SUBSTRACT FORCES FROM LINEAR FRICTION, BECAUSE IT DOESN'T ACTUATE TO ROTATION, ONLY TO
TRANSLATION
    %
**************!!!!!!!!!!!!!!!!!!*******************!!!!!!!!!!!!!!!!!!*******************!!!!!!!!!!!!!!!!!!*******************

    Qx = bR*(Fy + flin*vy*Fz);  % in case of + rotation around X axis, the ball rotates against the Y axis direction, so +
sign (we want actuating against friction force)
    Qy = -bR*(Fx + flin*vx*Fz); % minus sign, because in case of + rotation around Y axis, the ball rotates in the Y axis
direction (we want actuating against friction force)
    Qz = 0; % there is no actuate friction force, which actuates around Z axis
```

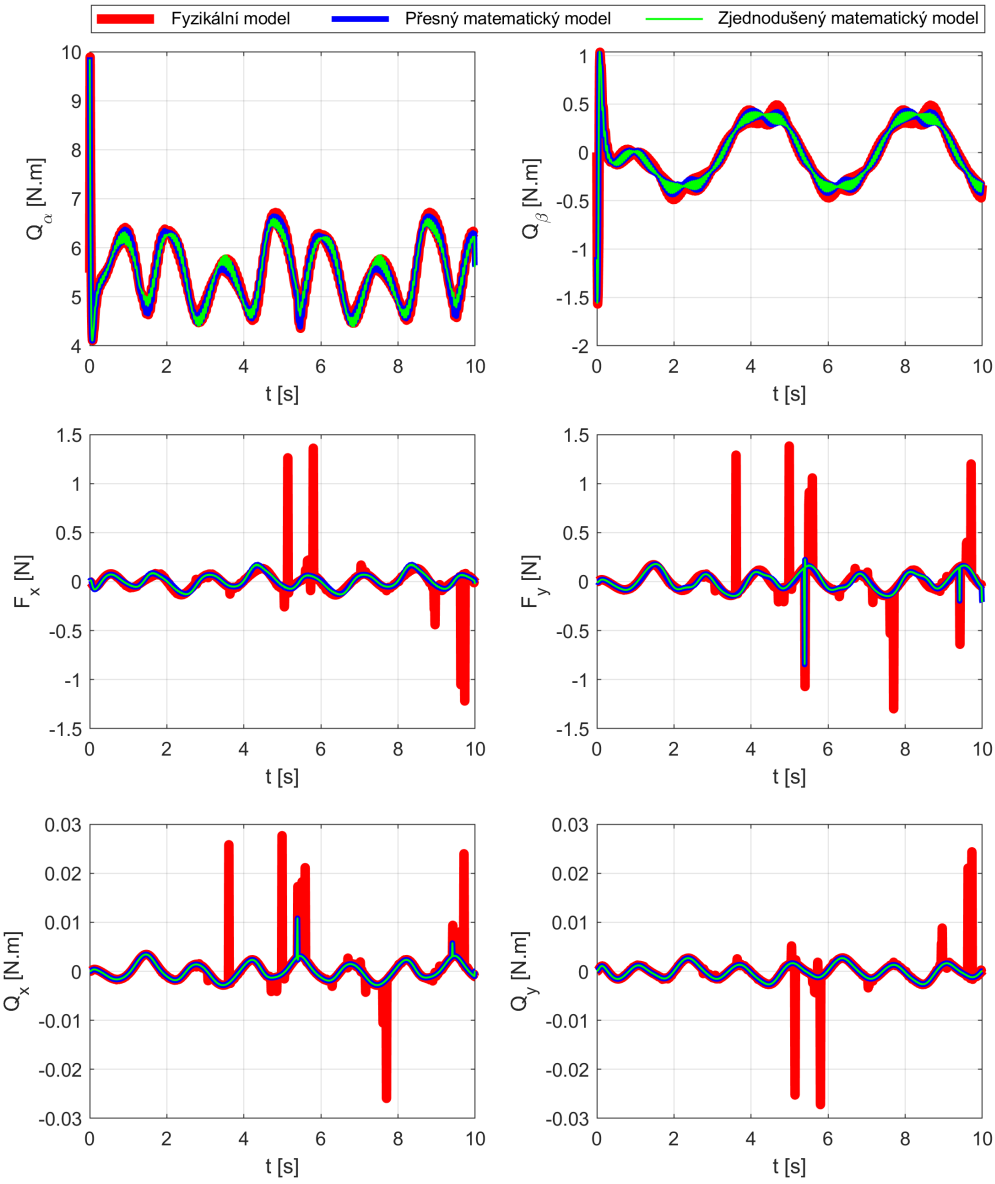
```
    % if the ball position is NOT higher than the wall, the wall pushes the ball into the plate area
elseif (pz < wallheight-bR/2)
    % ********************************************************************
    % This situation is for the ball in the corner of the plate
    % ********************************************************************

    % Support force from wal
    Fx = (abs(px)/px)*(Kpenw*(platex/2-bR-wallthick-abs(px))) - Dpenw*vx;
    Fy = (abs(py)/py)*(Kpenw*(platey/2-bR-wallthick-abs(py))) - Dpenw*vy;

    % Friction force from plate
    FxQ = friction(vxlin, vx, frstat, frkin, frtrh, flin)*Fz;
    FyQ = friction(vylin, vy, frstat, frkin, frtrh, flin)*Fz;

    % Friction force of wall
    pom = vzc*(abs(py)/py)-vx;  % with sides where the ball is, also direction of force is changed in case of rotation
around Z axis
    Fxw = friction(pom, vx, frstatw, frkinw, frtrh, flinw)*Fy;
    pom = -vzc*(abs(px)/px)-vy; % minus sign because of orientation of coordinate system in case of rotation around Z axis
    Fyw = friction(pom, vy, frstatw, frkinw, frtrh, flinw)*Fx;

    Qx = bR*(FyQ + flin*vy*Fz);    % rotation around X axis is based Fy friction from plate
    Qy = -bR*(FxQ + flin*vx*Fz);   % rotation around Y axis is based Fx friction from plate, signs are the same as in "if"
part
    Qz = bR*(Fxw + flinw*vx*Fy) - bR*(Fyw + flinw*vy*Fx);   % rotation around Z axis is based Fx and Fy frictions from wall

    % ********************************************************************
    % This situation is for the ball in the wall in X direction
    % ********************************************************************
    if (abs(px)>=(platex/2-bR-wallthick) && abs(py)<(platey/2-bR-wallthick))
        Fy = FyQ;       % force to Y direction is only from plate friction
        Qz = -bR*(Fyw + flinw*vy*Fx);   % torque around Z is only from X directrion wall

        % in case of the force from the wall will be bigger than the stiffness of the wall, the wall will break
        if (abs(Fx) >= (crushstr*touchar))
            breaked = 1;
            if (abs(Fx) < (crushstr*touchar)*1.5)
                fprintf('Wall is breaked with impact force Fx = %f\a N \n', abs(Fx));
            end
        end

        % ********************************************************************
        % This situation is for the ball in the wall in Y direction
        % ********************************************************************
    elseif (abs(py)>=(platey/2-bR-wallthick) && abs(px)<(platex/2-bR-wallthick))
        Fx = FxQ;       % force to X direction is only from plate friction
        Qz = bR*(Fxw + flinw*vx*Fy);    % torque around Z is only from Y directrion wall

        % in case of the force from the wall will be bigger than the stiffness of the wall, the wall will break
        if (abs(Fy) >= (crushstr*touchar))
            breaked = 1;
            if (abs(Fy) < (crushstr*touchar)*1.5)
                fprintf('Wall is breaked with impact force Fy = %f\a N \n', abs(Fy));
            end
        end

        % ********************************************************************
        % This situation is for the ball in the walls in a corner
        % ********************************************************************
    else
        % Suitable forces/torques are defined before this kind of "if" functions

        % in case of the force from the wall will be bigger than the stiffness of the wall, the wall will break
        if (abs(Fx) >= (crushstr*touchar) || abs(Fy) >= (crushstr*touchar))
            breaked = 1;
        end
    end

    % movement to the Z axis from the friction of the wall (first for the ball in -X side of the plate, next for +X side
with respect to
    % orientation of the rotations, translations, torques and movements)
    if (px<0)
        pom = -vxc-vz;
        Fzw = friction(pom, vz, frstatw, frkinw, frtrh, flinw)*Fx;
        % slow-down the rotation around Y and speed-up the translation in Z direction because of friction
        Qy = Qy + bR*(Fzw + flinw*vz*Fx);
        Fz = Fz + Fzw;
    else
        pom = vxc-vz;
        Fzw = friction(pom, vz, frstatw, frkinw, frtrh, flinw)*-Fx;

        % slow-down the rotation around Y and speed-up the translation in Z direction because of friction
        Qy = Qy - bR*(Fzw + flinw*vz*-Fx);
        Fz = Fz + Fzw;
    end
```

```
    % movement to the Z axis from the friction of the wall (first for the ball in -Y side of the plate, next for +Y side
with respect to
    % orientation of the rotations, translations, torques and movements)
    if (py<0)
        pom = -vyc-vz;
        Fzw = friction(pom, vz, frstatw, frkinw, frtrh, flinw)*Fy;
        % slow-down the rotation around X and speed-up the translation in Z direction
        Qx = Qx - bR*(Fzw + flinw*vz*Fy);
        Fz = Fz + Fzw;
    else
        pom = vyc-vz;
        Fzw = friction(pom, vz, frstatw, frkinw, frtrh, flinw)*-Fy;

        % slow-down the rotation around X and speed-up the translation in Z direction
        Qx = Qx + bR*(Fzw + flinw*vz*-Fy);
        Fz = Fz + Fzw;
    end

    % in case of the force from the wall will be bigger than the stiffness of the wall, the wall will break
    if (breaked == 1)
        Fx = 0;
        Fy = 0;
        Fz = 0;
        Qx = 0;
        Qy = 0;
        Qz = 0;
    end

    % if the ball position IS higher than the wall, the ball overleap the wall
else
    Fx = 0;
    Fy = 0;
    Fz = 0;
    Qx = 0;
    Qy = 0;
    Qz = 0;
end

% the simulation will be stopped, when the ball is in prohibited area (0.3 m under plate)
stop = 0;
if (pz < -0.3 && par(21) == 0)  % par(21) == 0 means that tbe ball is not disconnected by user
    stop = 1;
end

% prohibit rolling
if (par(20) == 1)
    Qx = 0;
    Qy = 0;
    Qz = 0;
end

% par(21) == 0 means that tbe ball is not disconnected by user. In other case, the ball have no connection with the plate
if (par(21) == 1)
    Fx = 0;
    Fy = 0;
    Fz = 0;
    Qx = 0;
    Qy = 0;
    Qz = 0;
end
end

%     % friction plot
%     frstat = 0.8;
%     frkin = 0.4;
%     frtrh = 1e-4;
%     flin = 1e-3;
%     x=-3e-4:1e-6:3e-4;
%     y = zeros(1,length(x));
%     for v = 1:length(x)
%         y(v) = friction(x(v), x(v), frstat, frkin, frtrh, flin);
%     end
%     plot(x,y,'r-','LineWidth',3);
%     ylim([-1 1]);
%     grid on;
%     hold on;
```

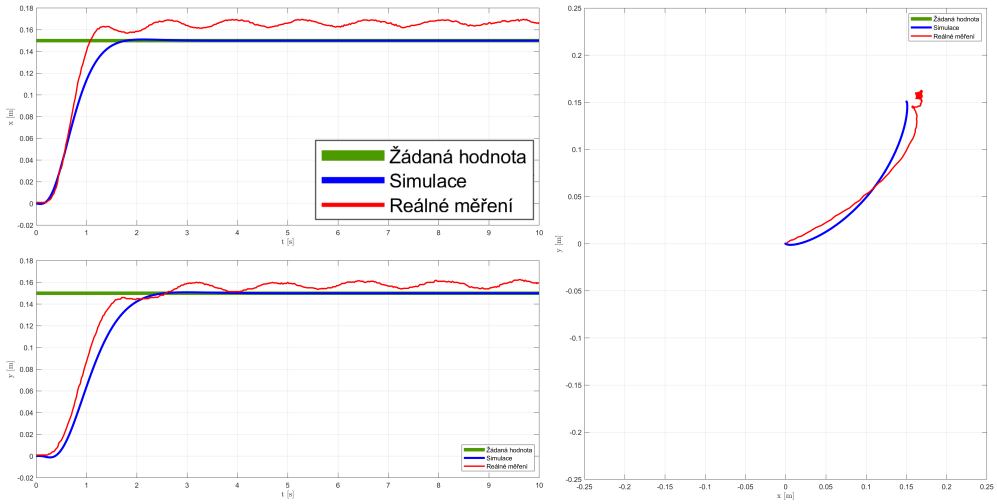
PŘÍLOHA G: MATEMATICKÝ + FYZIKÁLNÍ MODEL - TESTOVACÍ PRŮBĚH: KRUŽNICE



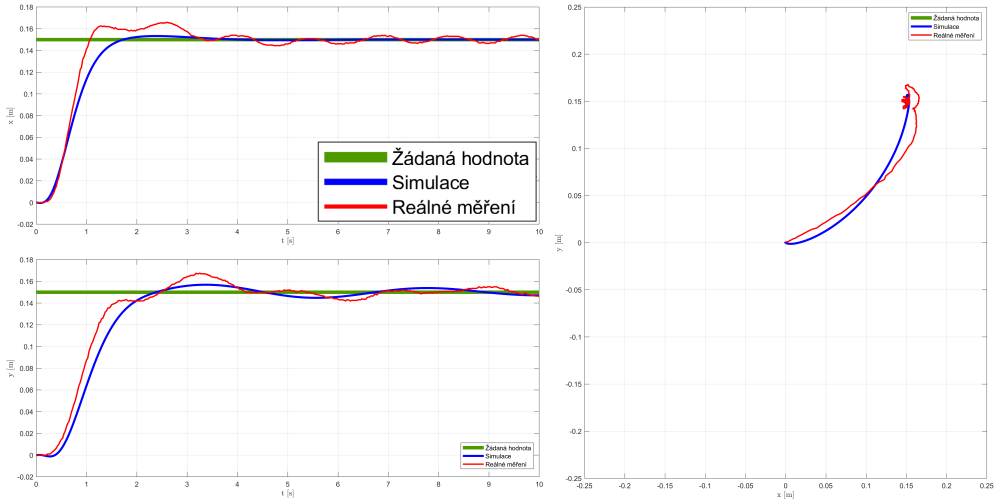
PŘÍLOHA H: MATEMATICKÝ + FYZIKÁLNÍ MODEL - TESTOVACÍ PRŮBĚH: ASTEROIDA



PŘÍLOHA I: REGULACE KULIČKY - BOD - RT

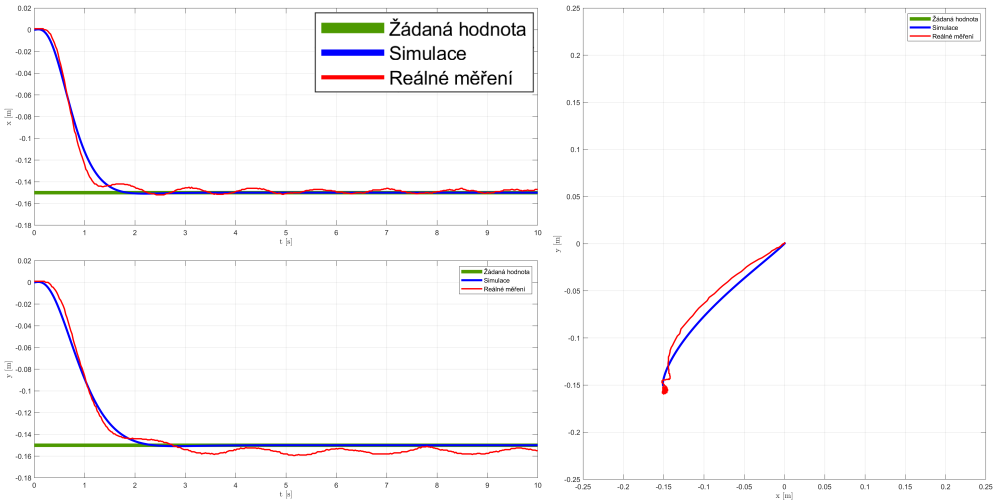


(a) PD regulátor

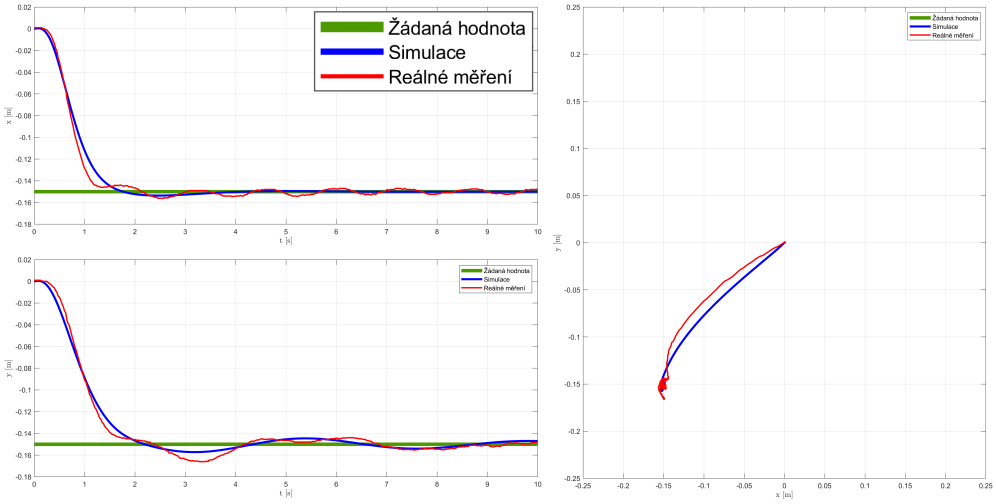


(b) PD(I) regulátor

PŘÍLOHA J: REGULACE KULIČKY - BOD - LB

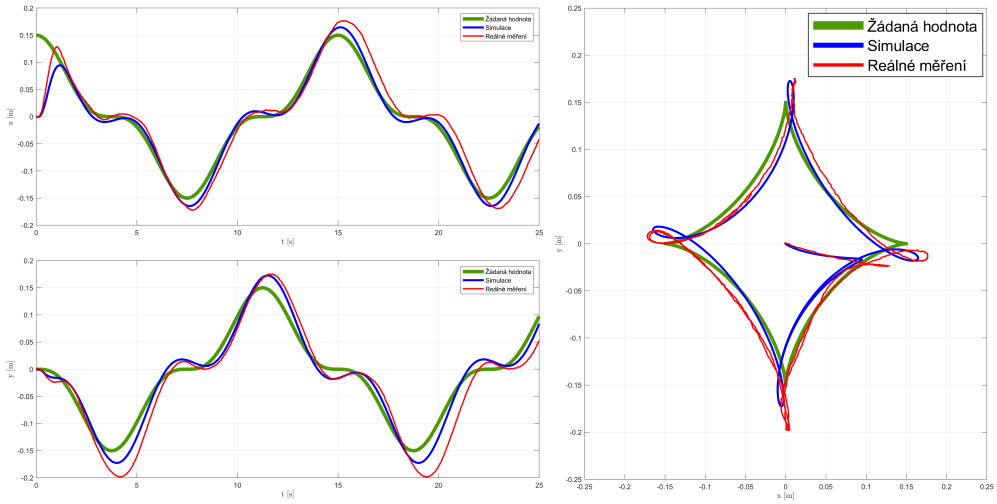


(a) PD regulátor

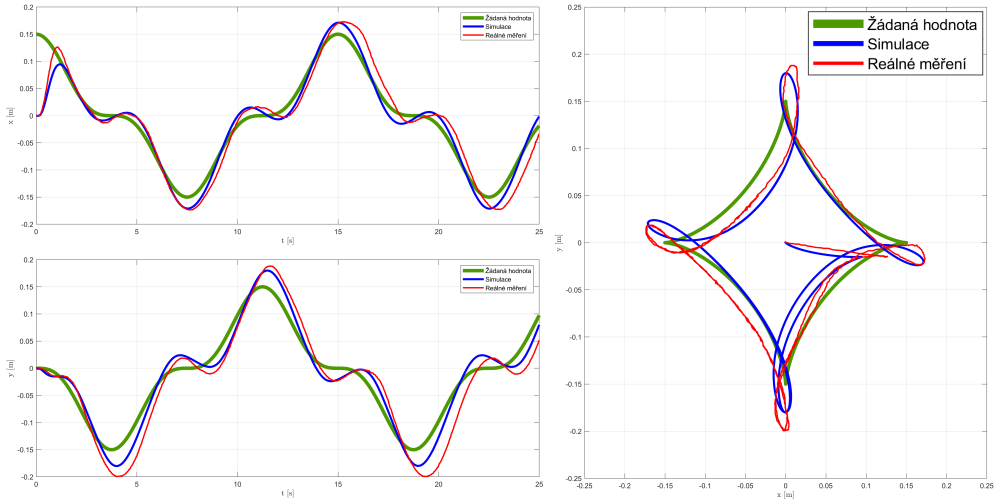


(b) PD(I) regulátor

PŘÍLOHA K: REGULACE KULIČKY - ASTEROIDA - T = 15 s

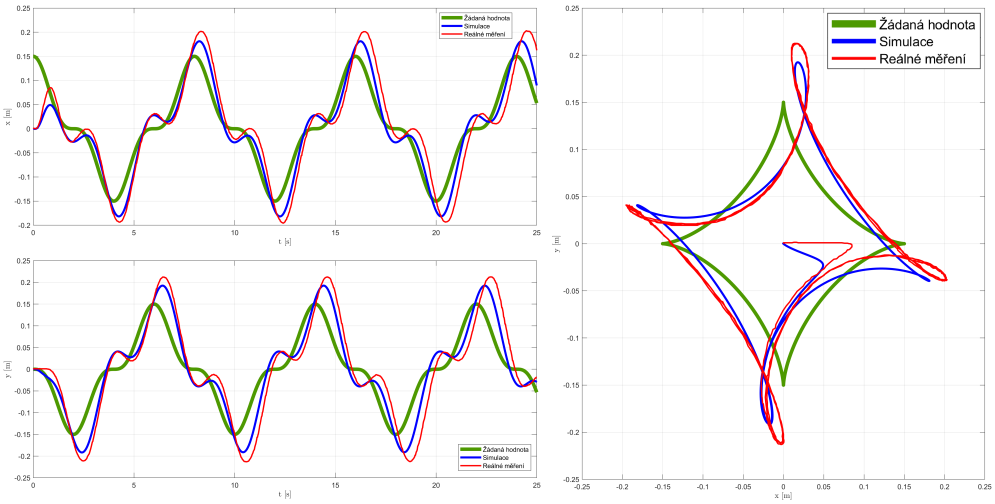


(a) PD regulátor

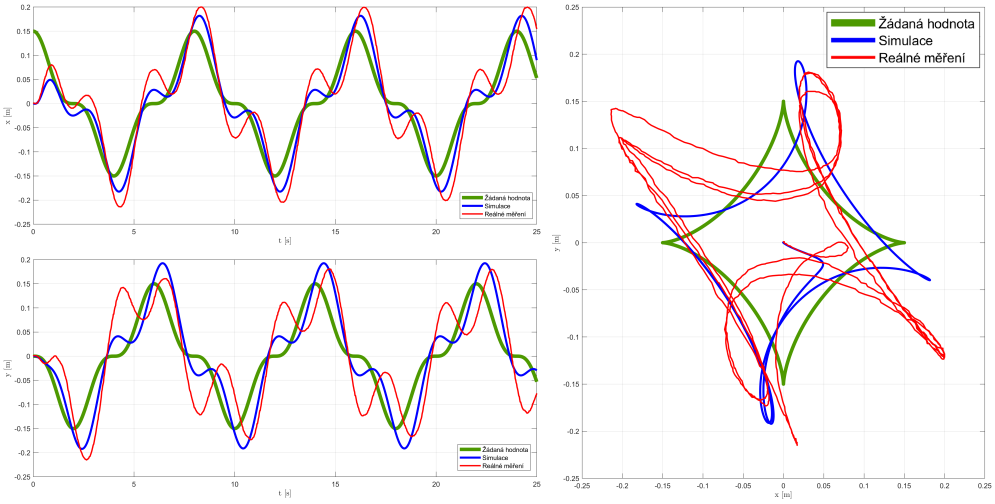


(b) PD(I) regulátor

PŘÍLOHA L: REGULACE KULIČKY - ASTEROIDA - T = 8 s

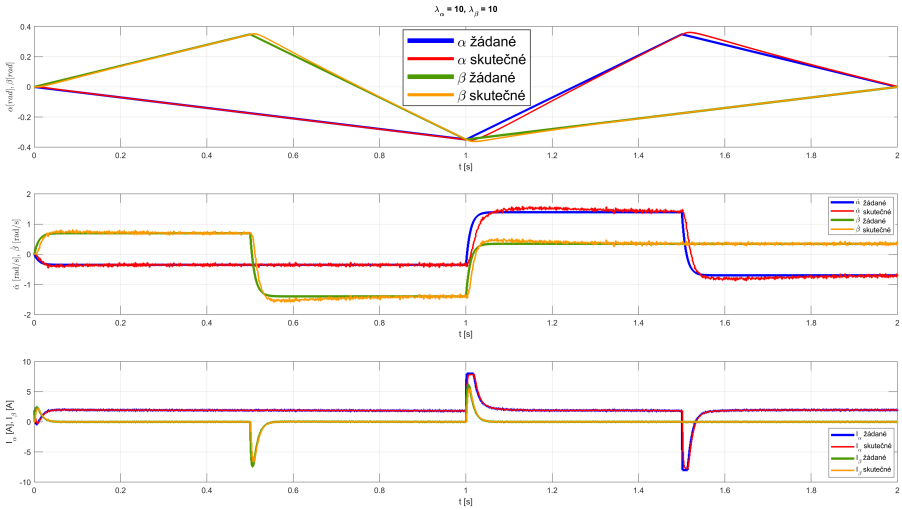


(a) PD regulátor

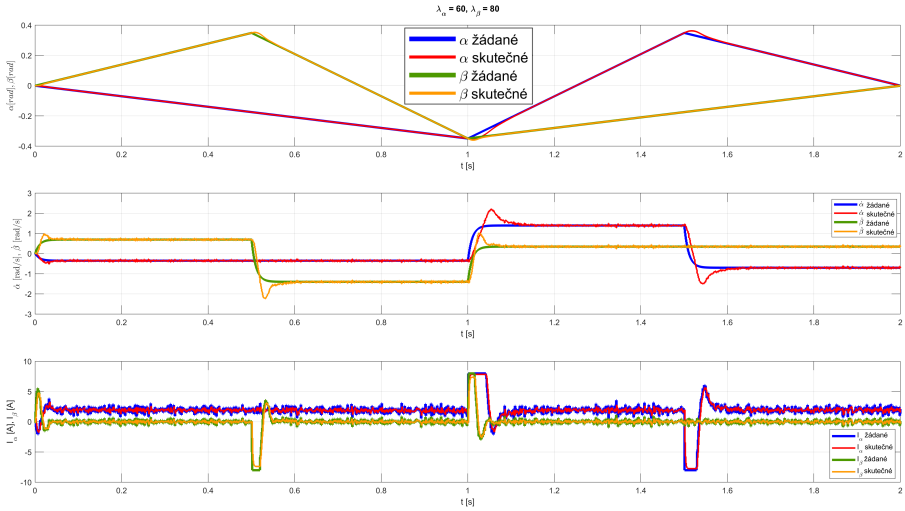


(b) PD(I) regulátor

PŘÍLOHA M: NASTAVENÍ ZPĚTNOVAZEBNÍCH ZESÍLENÍ

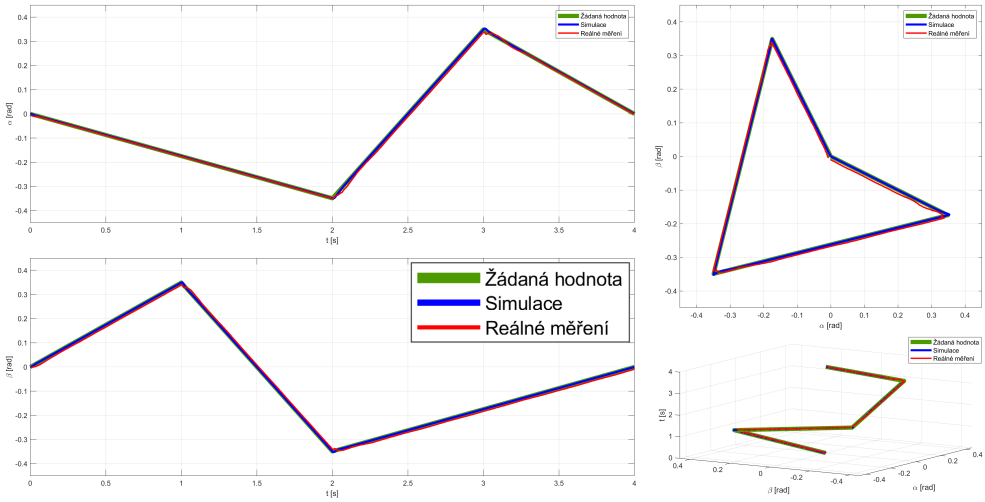


(a) Průběhy pro nízké hodnoty zpětnovazebních zesílení

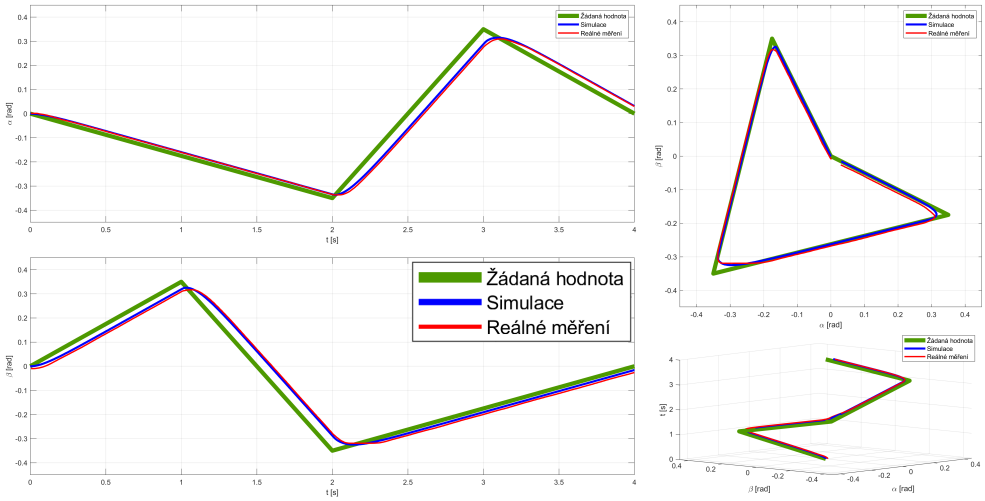


(b) Průběhy pro vysoké hodnoty zpětnovazebních zesílení

PŘÍLOHA N: REGULACE ROVINY - LINEÁRNÍ - ROZŠÍŘENÍ RVM

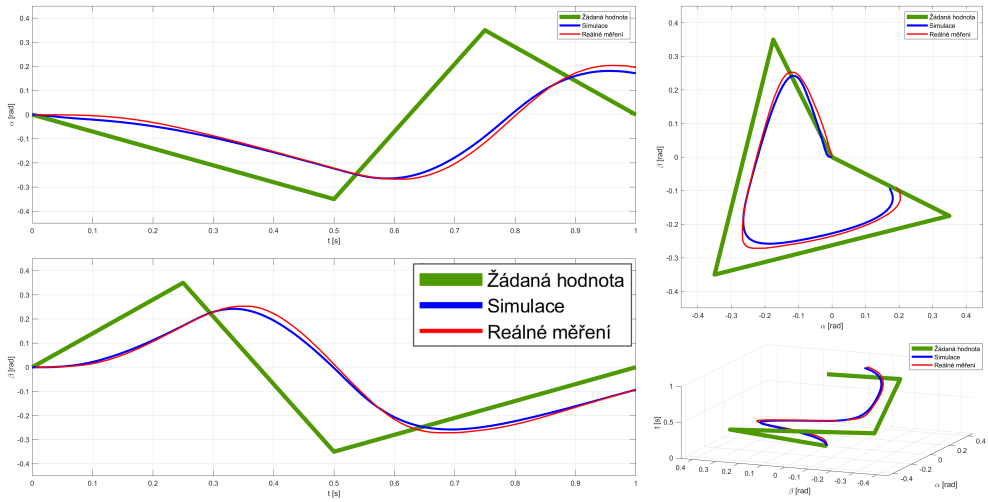


(a) Výpočet momentů - lineární - T = 4 s - bez integrátoru, bez filtrace

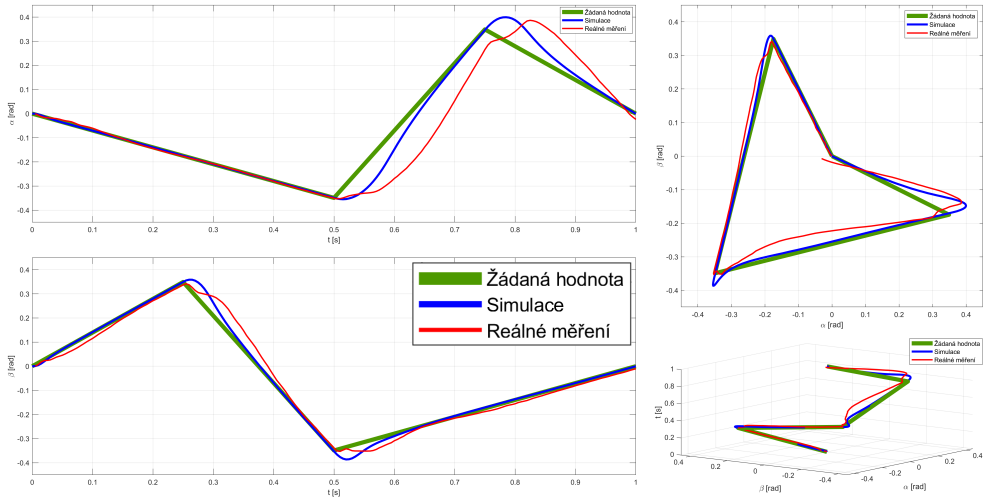


(b) Výpočet momentů - lineární - T = 4 s - s integrátorem, s filtrací

PŘÍLOHA O: REGULACE ROVINY - LINEÁRNÍ

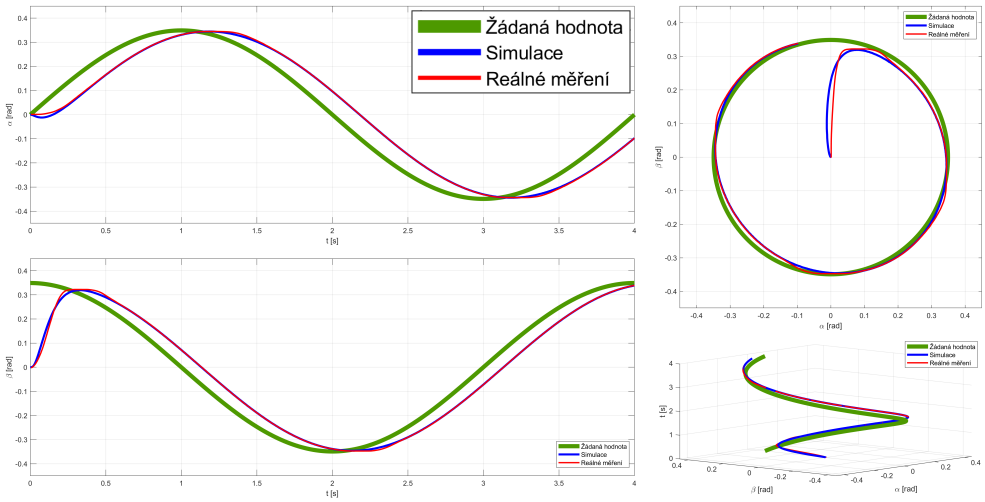


(a) Kaskáda - lineární - T = 1 s

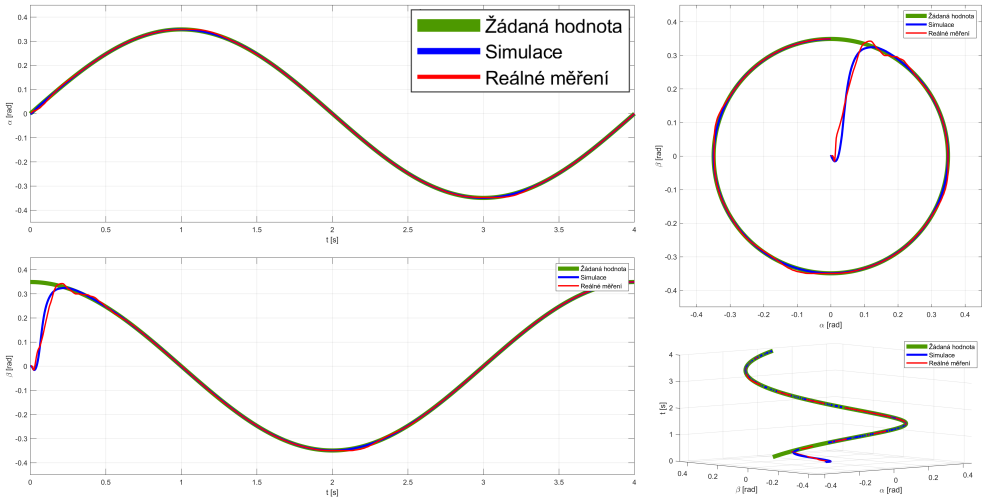


(b) Výpočet momentů - lineární - T = 1 s

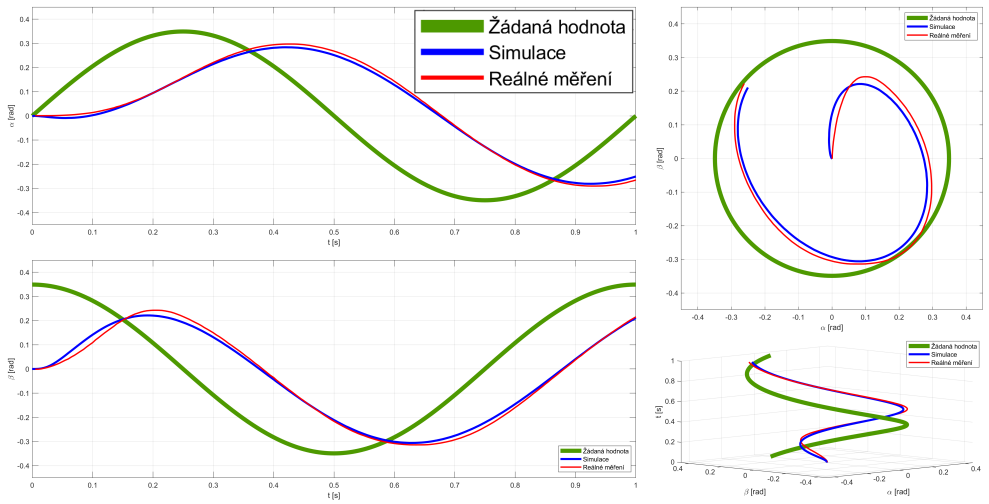
PŘÍLOHA P: REGULACE ROVINY - KRUŽNICE



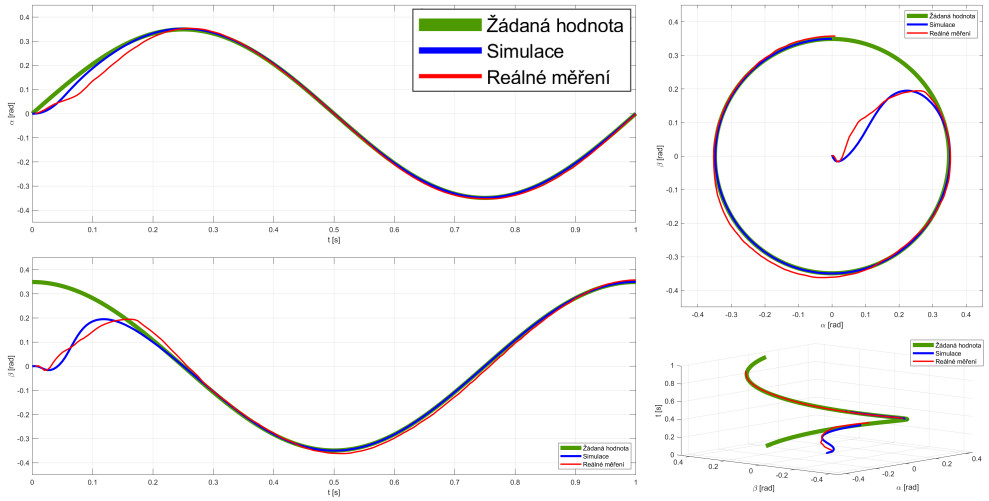
(a) Kaskáda - kružnice - T = 4 s



(b) Výpočet momentů - kružnice - T = 4 s

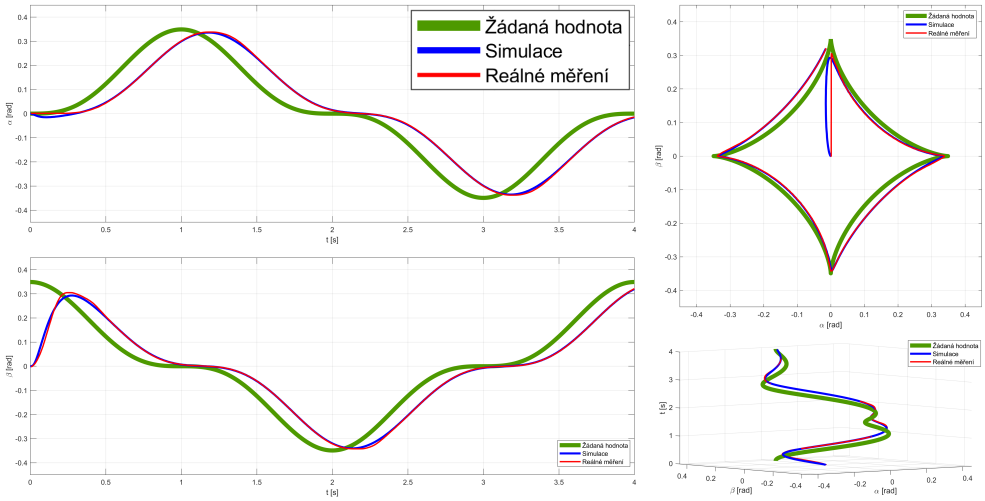


(a) Kaskáda - kružnice - T = 1 s

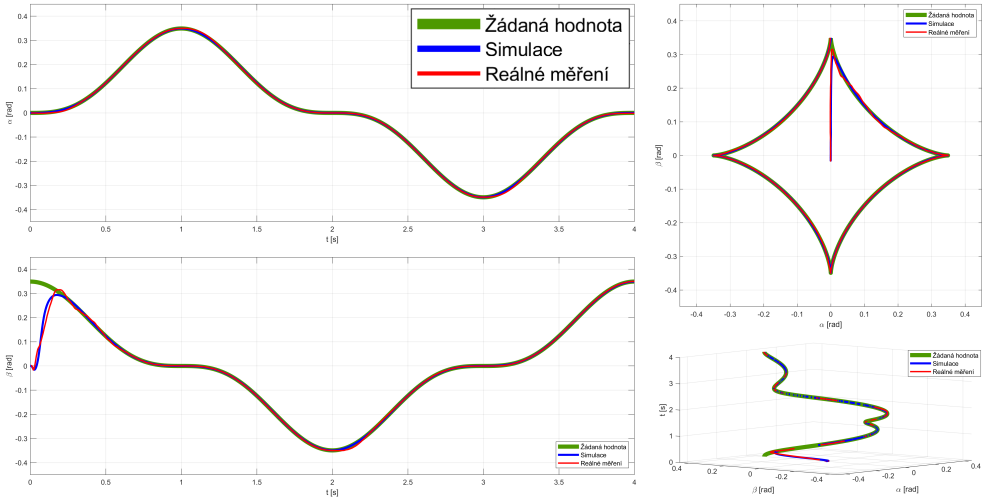


(b) Výpočet momentů - kružnice - T = 1 s

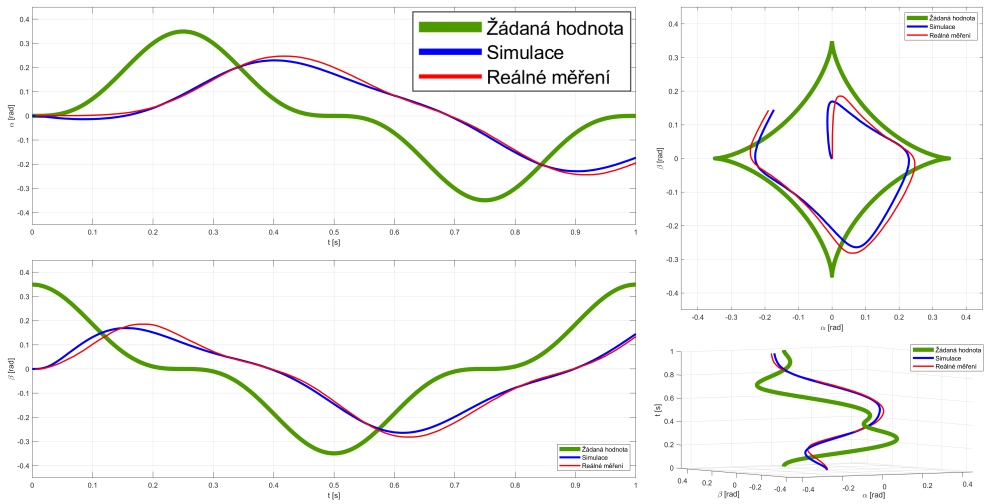
PŘÍLOHA Q: REGULACE ROVINY - ASTEROIDA



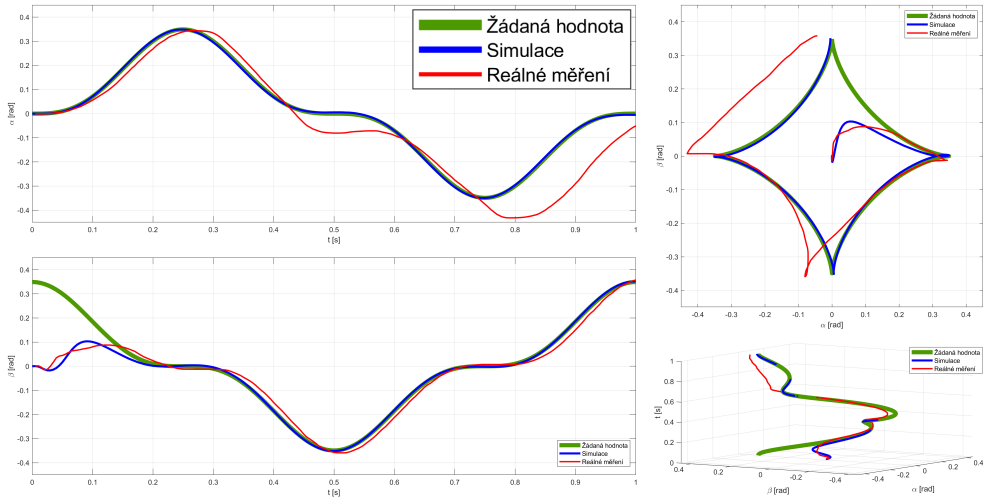
(a) Kaskáda - asteroida - T = 4 s



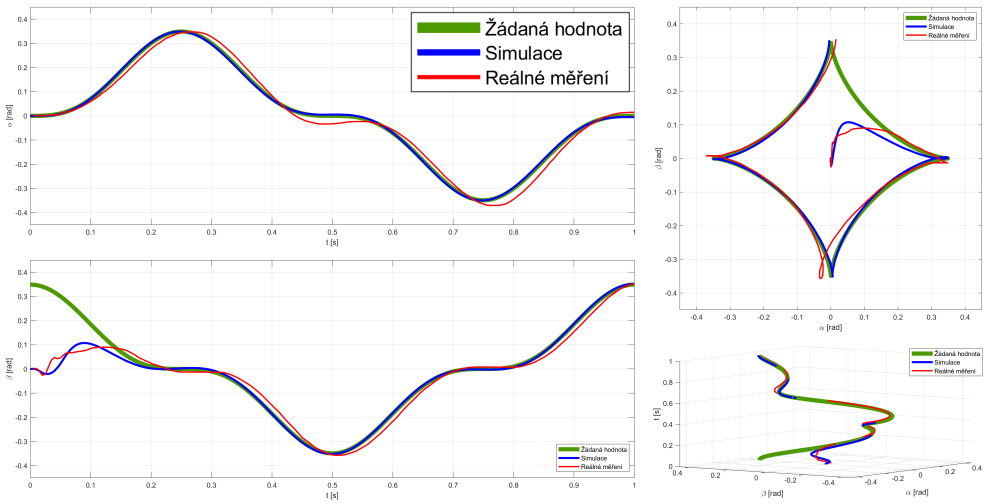
(b) Výpočet momentů - asteroida - T = 4 s



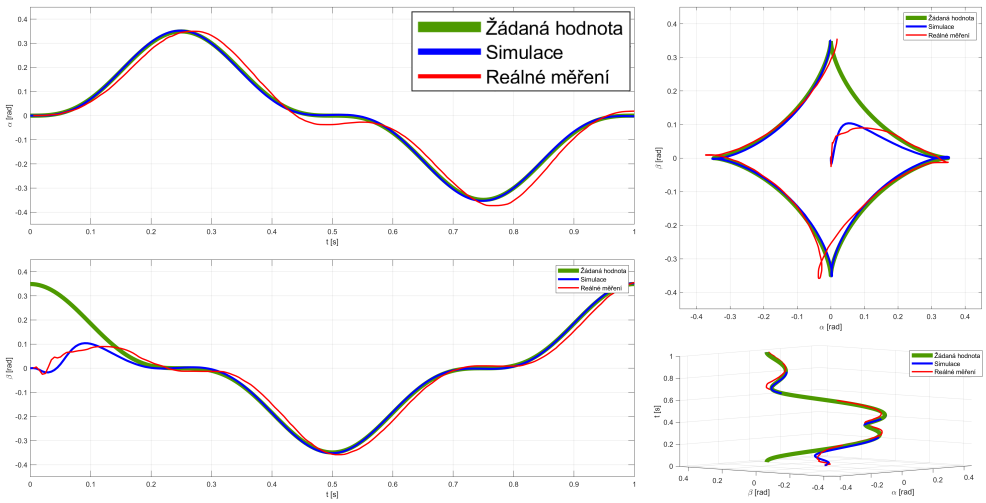
(c) Kaskáda - asteroida - T = 1 s



(d) Výpočet momentů - asteroida - T = 1 s

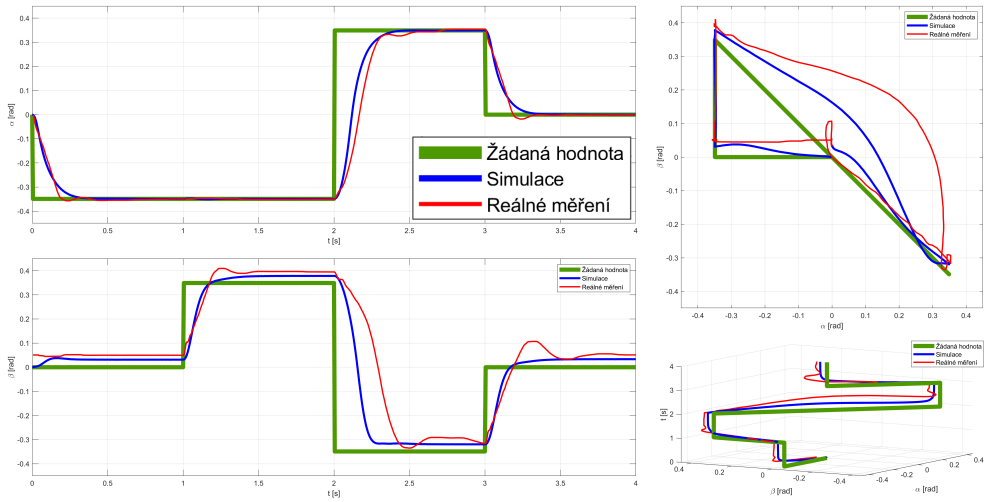


(e) Výpočet momentů - asteroida - T = 1 s - I_{max} = 16 A - zjednodušený model

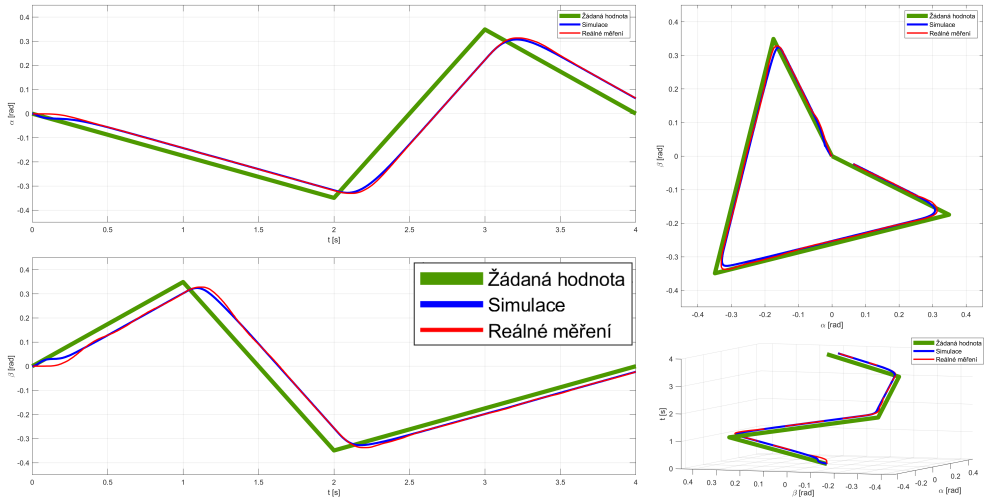


(f) Výpočet momentů - asteroida - T = 1 s - I_{max} = 16 A - přesný model

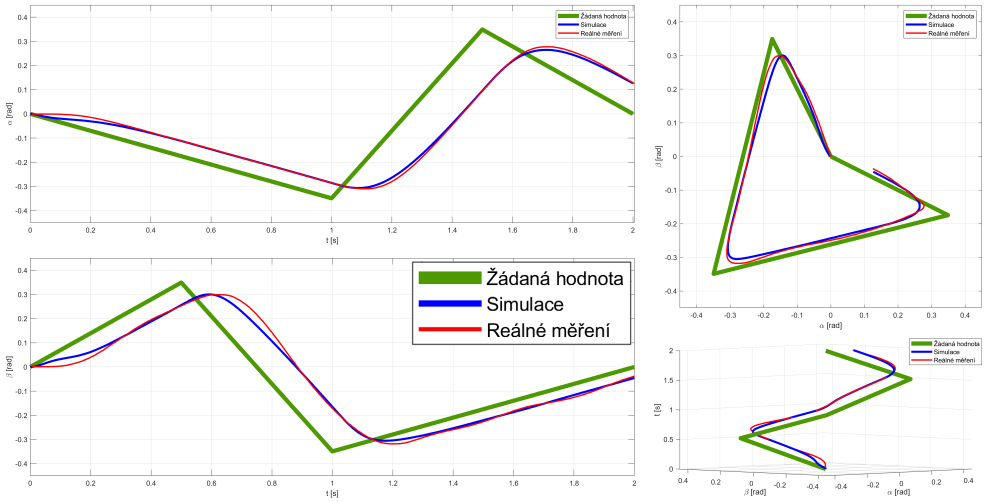
PŘÍLOHA R: REGULACE ROVINY - ZÁVAŽÍ



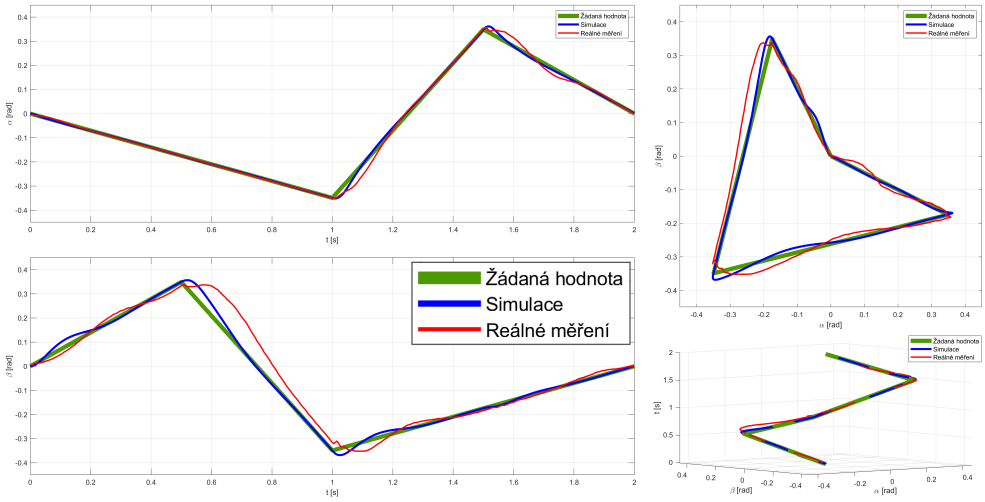
(a) Výpočet momentů - skoky - závaží - T = 4 s - $Hyst_{Int} = 1$



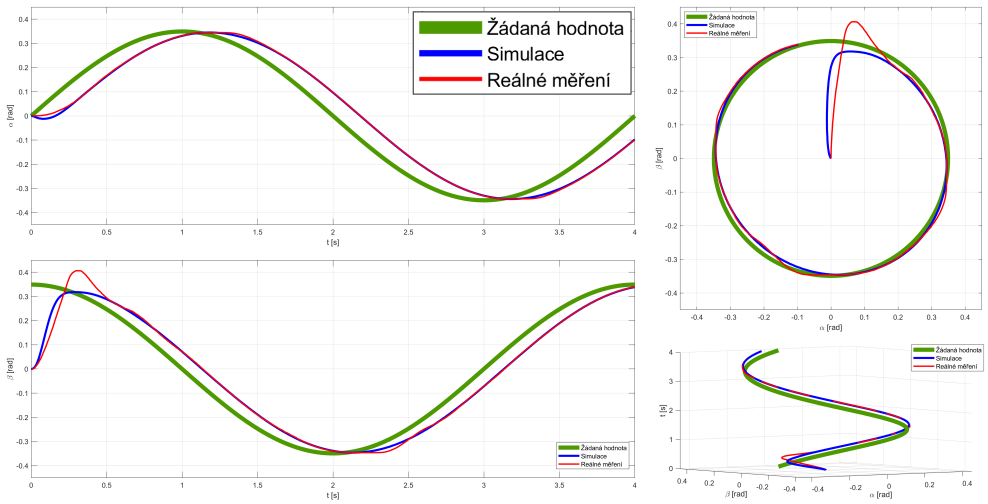
(b) Kaskáda - lineární - závaží - T = 4 s



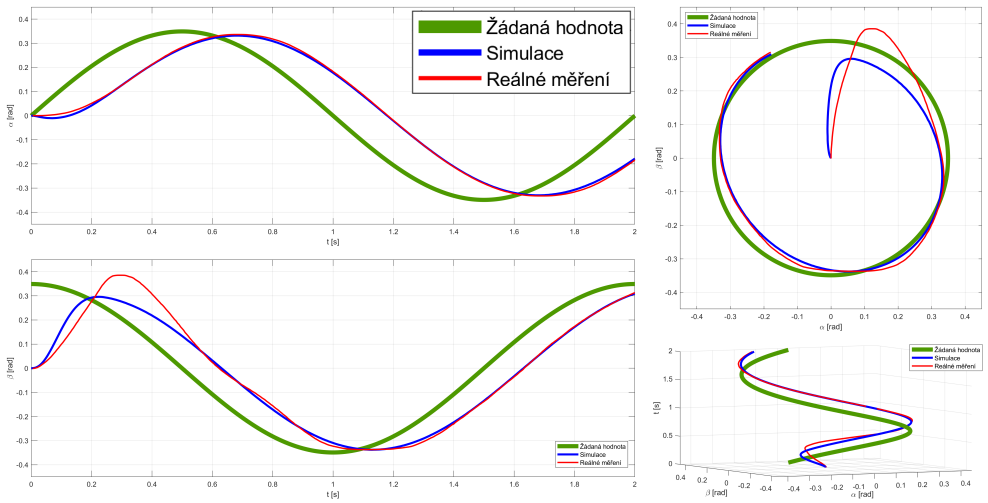
(c) Kaskáda - lineární - závaží - T = 2 s



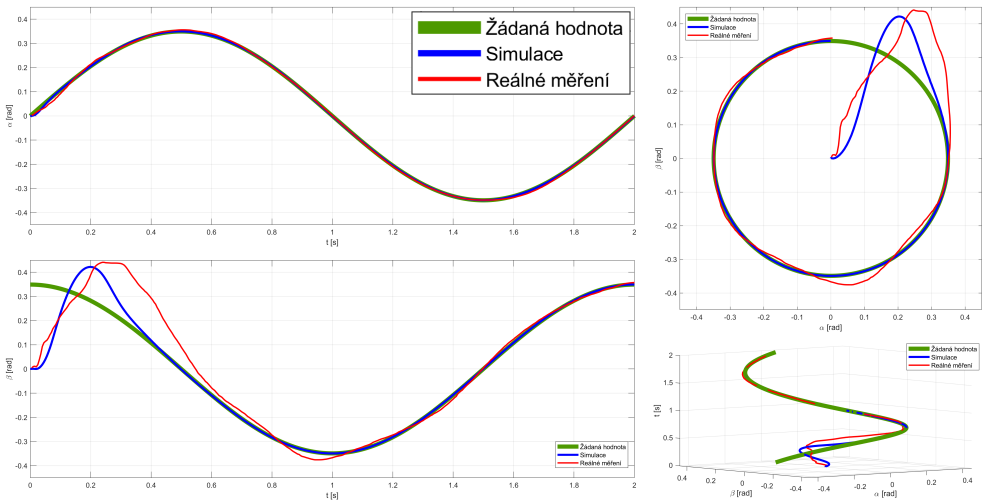
(d) Výpočet momentů - lineární - závaží - T = 2 s



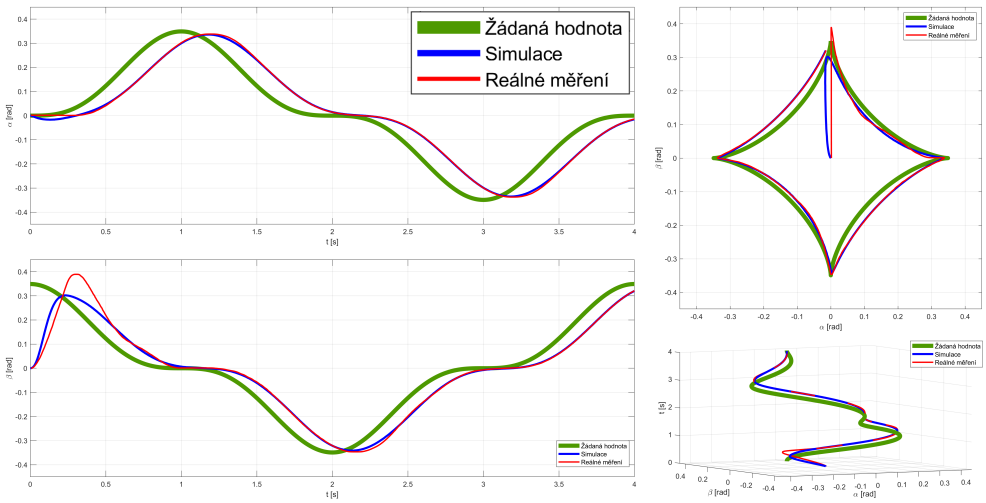
(e) Kaskáda - kružnice - závaží - T = 4 s



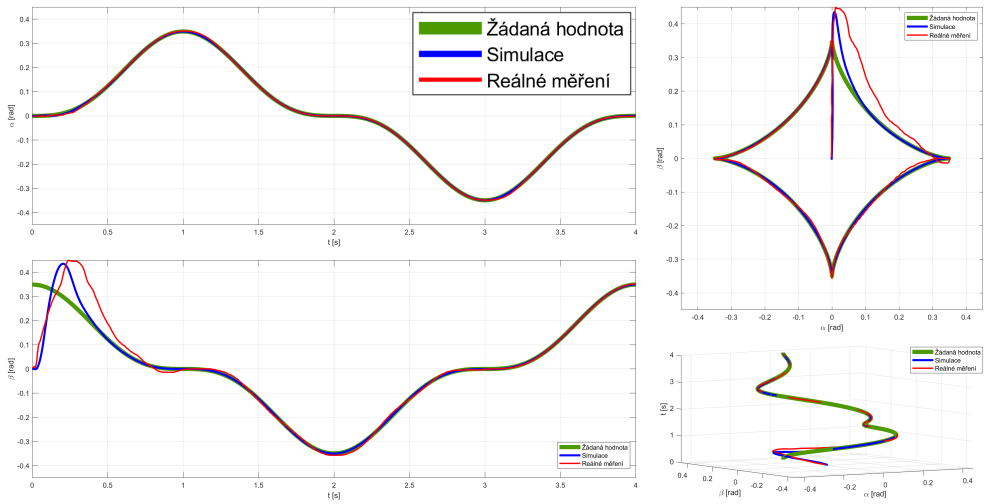
(f) Kaskáda - kružnice - závaží - T = 2 s



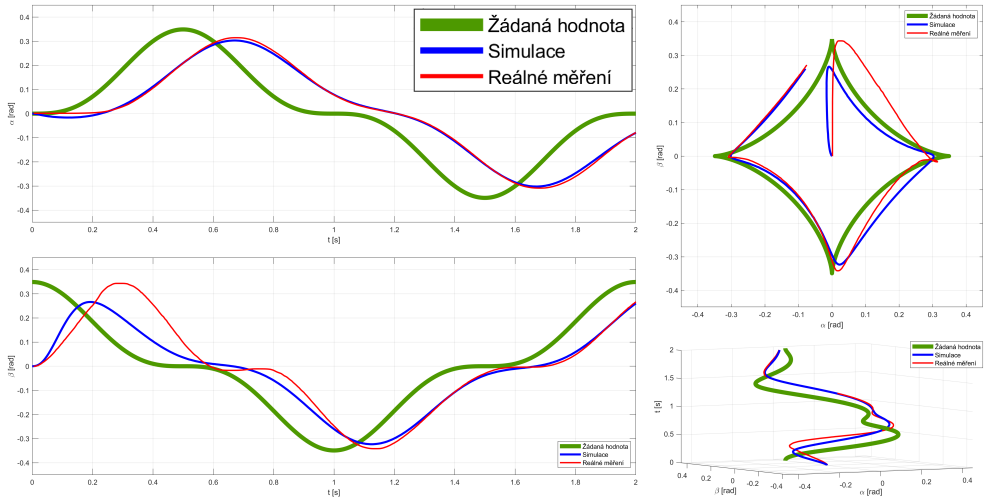
(g) Výpočet momentů - kružnice - závaží - T = 2 s



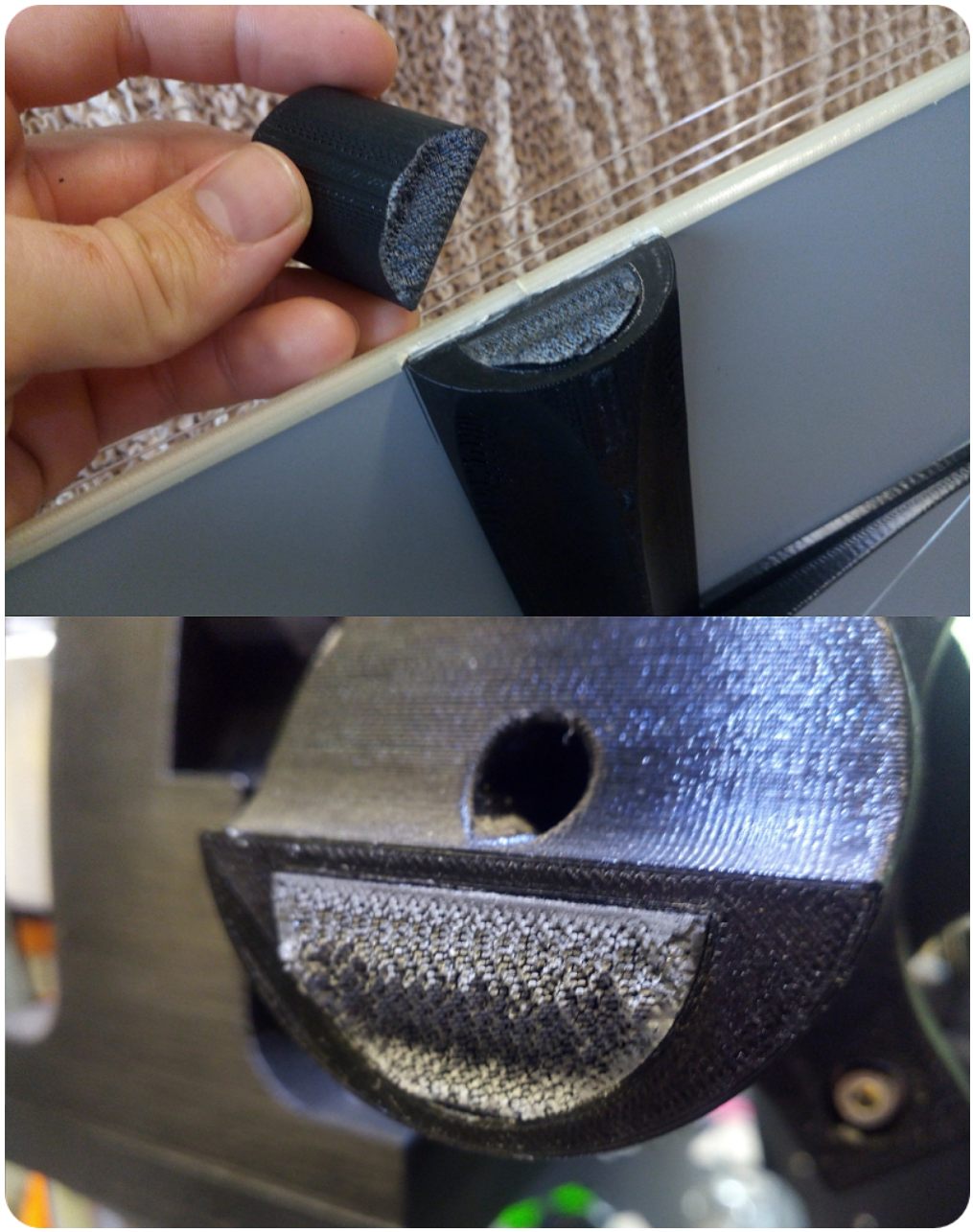
(h) Kaskáda - asteroida - závaží - T = 4 s



(i) Výpočet momentů - asteroida - závaží - T = 4 s

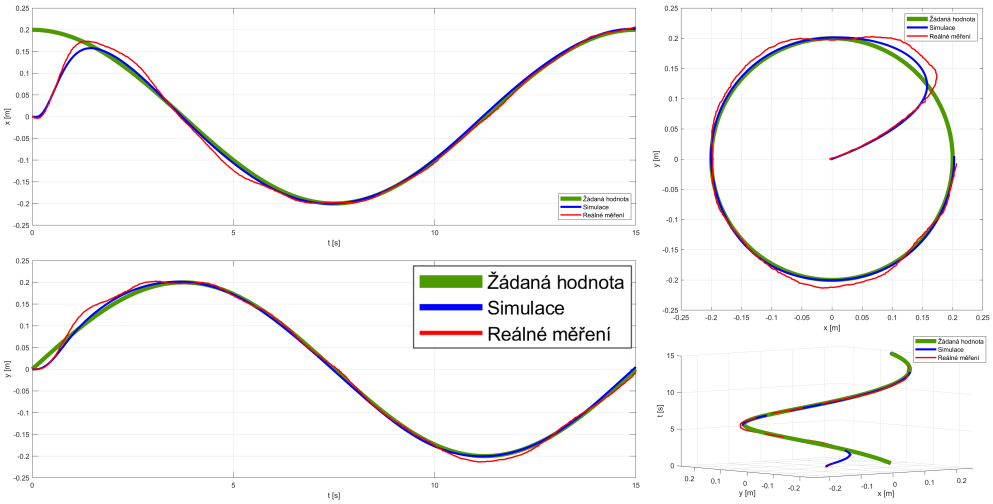


(j) Kaskáda - asteroida - závaží - T = 2 s

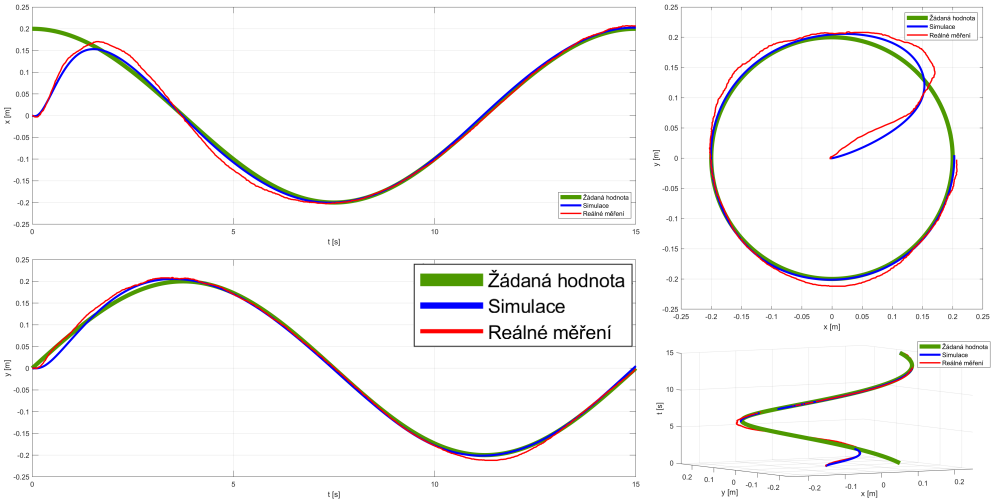


(k) Zlomení ramene roviny při vysoké dynamice pohybu

PŘÍLOHA S: REGULACE KULIČKY - RK VS RVM

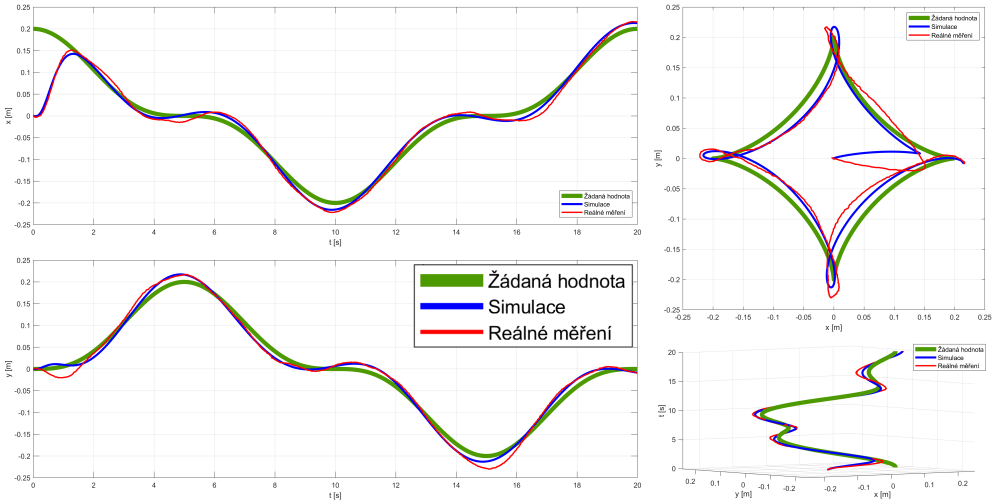


(a) Regulace kuličky s RK - kružnice - T = 15 s

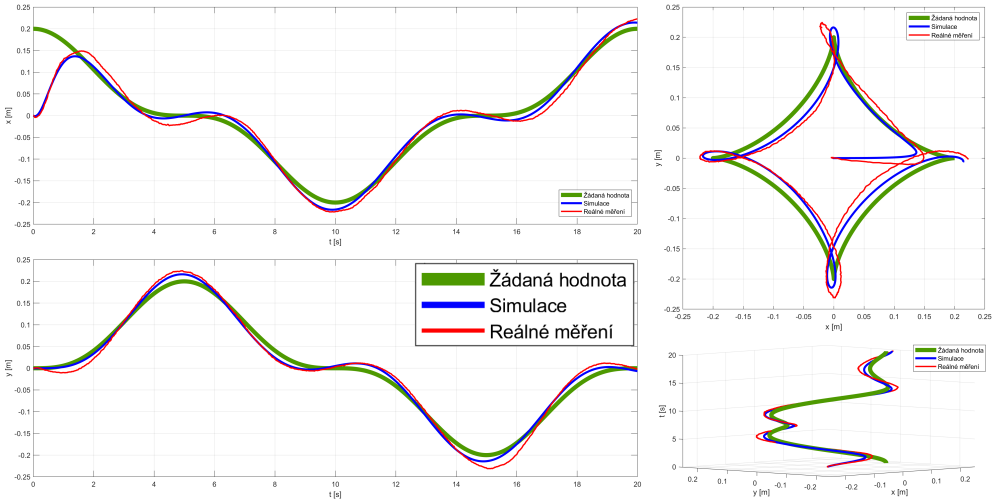


(b) Regulace kuličky s RVM - kružnice - T = 15 s

,



(c) Regulace kuličky s RK - asteroida - T = 20 s



(d) Regulace kuličky s RVM - asteroida - T = 20 s