



TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií ■

Rychlé heuristické metody numerického řešení úlohy inverzní kinematiky

Disertační práce

Studijní program: P2612 – Elektrotechnika a informatika
Studijní obor: 2612V045 – Technická kybernetika

Autor práce: **Ing. Jan Čejka**
Školitel: doc. Ing. Josef Černohorský, Ph.D.



Fast heuristic methods of numerical solution of inverse kinematics

Doctoral thesis

Study programme: P2612 – Electrical Engineering and Informatics
Study branch: 2612V045 – Technical Cybernetics

Autor: **Ing. Jan Čejka**
Supervisor: doc. Ing. Josef Černohorský, Ph.D.



Prohlášení

Byl jsem seznámen s tím, že na mou disertační práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé disertační práce pro vnitřní potřebu TUL.

Užiji-li disertační práci, nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Disertační práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé disertační práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

Podpis:

Abstrakt

Při zkoumání rozsáhlých robotických soustav pro potřeby firmy ŠKODA AUTO a.s. byl vytvořen software, který je zaměřen na kontrolu robotů a robotových standardů používaných v koncernu Volkswagen AG. Součástí je i matematická knihovna, která řeší několik základní inženýrských úloh. Jedná se o přímou a inverzní kinematickou úlohu v robotice a o řešení soustav lineárních rovnic více proměnných. Při zkoumání numerických metod v rámci programování uvedených úloh byly vyvinuty dva zcela nové algoritmy. Jedná se o metodu relaxace úhlu, která je určena pro řešení soustav lineárních rovnic. Může být také nepřímo použita pro řešení inverzní kinematické úlohy v robotice. Druhý algoritmus je přímo určen pro řešení inverzní kinematické úlohy v robotice. V tomto případě se jedná o metodu relaxace délky.

Metoda relaxace úhlu je diskutována z hlediska základního principu a numerických vlastností při výpočtu soustav lineárních rovnic. Je zkoumána rychlost konvergence, závislost na číslu podmíněnosti a schopnost řešit obecné soustavy lineárních rovnic pro různé podoby matice soustavy. V rámci jednotlivých experimentů se metoda porovnává s výsledky známých numerických metod. Ukazuje se, že metoda relaxace úhlu konverguje k výsledkům podobným jako Mooreova-Penroseova pseudoinverze i v případě singulární matice soustavy. Tato vlastnost je vhodná pro řešení inverzní kinematické úlohy v robotice, protože se při výpočtu matice soustavy dynamicky mění v závislosti na postavení kinematické struktury robota.

Aby bylo možné metodu relaxace úhlu použít pro výpočet inverzní kinematické úlohy v robotice, vychází práce z přístupů, které převádí tuto problematiku do podoby soustav lineárních rovnic. Jelikož inverzní kinematická úloha v robotice vede na soustavy nelineárních rovnic, jedná se o přístupy, které linearizují řešení ve vybraném pracovním bodě pomocí Jacobiho matice, tzn. Newtonova metoda, inverze Jacobiho matice a metoda Levenberg-Marquardt. Všechny tři přístupy jsou odzkoušeny na kinematické struktuře planárního manipulátoru a robota KUKA KR210 R2700 EXTRA. Vzniklé soustavy lineárních rovnic jsou řešeny standardní cestou pomocí Mooreovy-Penroseovy pseudoinverze a zároveň metodou relaxace úhlu. Výsledky obou přístupů jsou vyhodnoceny a porovnány.

Na závěr práce je diskutována metoda relaxace délky, kterou lze zařadit do skupiny heuristických metod. Metoda je popsána z hlediska jejího principu a porovnána se známými heuristickými metodami CCD a FABRIK.

Klíčová slova: soustavy lineárních rovnic, inverzní kinematická úloha, pseudoinverze, robotika

Abstract

As part of a research of large robotic systems for the needs of SKODA AUTO, a.s. company a inspectional software was developed. The software focuses on a control of robots and programming standards used by Volkswagen AG. The software includes a mathematical library which solves several basic engineering tasks. It is a problem of forward and inverse kinematics in robotics and a solution of systems of linear equations. This work introduces two completely new algorithms which were developed during the programming of the mathematical library. It is angle relaxation method which is designed for solving of systems of linear equations. It can also be used indirectly to solve inverse kinematics in robotics. The second algorithm is designed directly for solving of inverse kinematics in robotics. In this case the work talks about length relaxation method.

The angle relaxation method is discussed from the point of view of basic principle and numerical properties. A speed of convergence, a dependence on the condition number and an ability to solve general systems of linear equations for different forms of matrices were examined. Results of angle relaxation method were compared with results of known numerical methods. It appears that the angle relaxation method converges to results similar to results of Moore-Penrose pseudoinverse. This property is suitable for the solving of inverse kinematics in robotics because the system matrix is dynamically changed during a calculation depending on a position of the kinematic structure of the robot.

In order to use the angle relaxation method to calculate the inverse kinematics in robotics, the work is based on approaches which transform the problem into systems of linear equations. The inverse kinematics in robotics leads primarily to systems of nonlinear equations. For this reason approaches were chosen which linearize the equation using a Jacobian matrix, i.e. Newton's method, inverse Jacobian method and Levenberg-Marquardt method. All three approaches were tested on the kinematic structure of the planar manipulator and the robot KUKA KR210 R2700 EXTRA. Systems of linear equations were solved by a standard way using Moore-Penrose pseudoinverse and at the same time by the angle relaxation method. Results of both approaches were evaluated and compared.

At the end of this work the length relaxation method is discussed which can be included in the group of heuristic methods. The length relaxation method is described from the point of view of basic principle and compared with known heuristic methods CCD and FABRIK.

Key words: systems of linear equations, inverse kinematics, pseudoinverse, robotics

Obsah

Použité značky a symboly.....	7
Seznam obrázků	8
Seznam tabulek	11
1 Úvod	12
1.1 Zavedení matematických objektů	16
2 Metoda relaxace úhlu	19
2.1 Popis metody relaxace úhlu	19
2.2 Konvergence a stabilita metody relaxace úhlu	24
2.3 Maticový zápis algoritmu relaxace úhlu.....	27
2.4 Vlastnosti metody relaxace úhlu	29
3 Výpočet inverzní kinematické úlohy v robotice.....	40
3.1 Kinematická struktura a její popis.....	40
3.2 Přímá kinematická úloha v robotice.....	43
3.3 Rozměrová analýza	46
3.4 Inverzní kinematická úloha v robotice	49
3.5 Porovnání vybraných numerických metod	52
3.6 Praktická aplikace	69
4 Metoda relaxace délky	72
4.1 Popis metody relaxace délky	72
4.2 Porovnání vybraných heuristických metod	79
5 Závěr	80
Použitá literatura	83
Seznam publikací autora	86

Použité značky a symboly

$\mathbb{R}, \mathbb{C}, \mathbb{N}$	obor reálných, komplexních, přirozených čísel
\mathbb{R}^n	aritmetický vektorový prostor tvořený n-ticemi reálných čísel
V	vektorový prostor
x	vektor
x^R	relaxovaný vektor
\hat{x}	normovaný vektor
a, b, \dots	skalár
o	nulový vektor
O	nulová matice
A	matice soustavy
P	matice nevlastního řešení soustavy
r	reziduum
p	sloupec matice nevlastního řešení soustavy
\mathbb{E}_n	eukleidovský prostor konečné dimenze n
P_1, P_2, \dots	body
$\oplus +$	sčítání vektorů
$\odot \cdot$	násobení vektorů
α, β, \dots	číslo z tělesa
M	množina
$\kappa(A)$	číslo podmíněnosti matice
A^{-1}	inverze matice
A^+	pseudoinverze matice
MIMO	Multiple Input Multiple Output
SISO	Single Input Single Output
CCD	Cyclic Coordinate Descent
FABRIK	Forward And Backward Reaching Inverse Kinematics
RLXA, RLX	Relaxace úhlu

Seznam obrázků

Obr. 1: Robotová soustava - dovářecí stanice	13
Obr. 2: Software SKODA TOOL.....	14
Obr. 3: Relaxace úhlu	19
Obr. 4: Geometrická interpretace soustavy lineárních rovnic pomocí vektorů.....	20
Obr. 5: Geometrická interpretace převodu nevlastního řešení soustavu na řešení soustavy vlastní.....	22
Obr. 6: Základní operace algoritmu relaxace úhlu	24
Obr. 7: Trojúhelníková nerovnost	24
Obr. 8: Velikost rezidua v k-tém iteračním kroku s náhodnou proměnnou σ a bez náhodné proměnné σ	26
Obr. 9: Řešení soustavy $\mathbf{x} = (1,2,3)^T$ v k-tém iteračním kroku s náhodnou proměnnou σ a bez náhodné proměnné σ	26
Obr. 10: Metoda relaxace úhlu v maticové podobě.....	28
Obr. 11: Čas potřebný pro vyřešení soustavy lineárních rovnic vybraných numerických metod.....	33
Obr. 12: Vliv čísla podmíněnosti matice soustavy na rychlost konvergence metody relaxace úhlu, $n = 10$	33
Obr. 13: Vliv velikosti soustavy lineárních rovnic na rychlosti konvergence metody relaxace úhlu, $\kappa(\mathbf{A}) = 1$	34
Obr. 14: Velikost rezidua v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic s nekonečně mnoho řešeními.....	36
Obr. 15: Řešení soustavy v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic s nekonečně mnoho řešeními	37
Obr. 16: Norma $\ \mathbf{x}\ _e$ pro sto náhodně generovaných soustav lineárních rovnic s nekonečně mnoho řešeními	37
Obr. 17: Velikost rezidua v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic bez řešení	38
Obr. 18: Řešení soustavy v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic bez řešení.....	38
Obr. 19: Norma $\ \mathbf{b} - \mathbf{A} \cdot \mathbf{x}\ _e$ pro sto náhodně generovaných soustav lineárních rovnic bez řešení....	39
Obr. 20: Přehled kinematických vazeb (R – Rotace, T – Translace).....	40
Obr. 21: Příklad sériového a paralelního uspořádání kinematické struktury (V – Vazba, S – Spoj) ...	41
Obr. 22: Pracovní prostor kinematické struktury dle typu robota	41
Obr. 23: Rotace kolem souřadného systému popsána parametry $[\alpha \beta \gamma]$	42
Obr. 24: Přímá a inverzní kinematická úloha v robotice	43
Obr. 25: Poloha koncového bodu angulárního průmyslového robota \mathbf{T}_R pro jednotlivé úhly natočení os θ_1 až θ_6	43

Obr. 26: Kinematická struktura planárního manipulátoru	44
Obr. 27: Příklady sériových robotů a manipulátorů, 2 osy - kartézský manipulátor, 3 osy – SCARA robot Yamaha, 4 osy – paletizační robot Fanuc, 5 os – robot R12 ST Robotics, 6 os – robot KR210 R2700 EXTRA KUKA, 7 os – LBR IIWA KUKA, 8 os – robot FARO Quantum, n os – nerealizovaná studie robotického hada.....	48
Obr. 28: Metody řešení inverzní kinematické úlohy v robotice – jednoduché rozdělení	50
Obr. 29: Rozdíl žádané a aktuální polohy koncového bodu kinematické struktury. Pro velké r je nutné rozdělit dráhu na několik menších úseků kvůli nelinearitě.....	51
Obr. 30: Planární manipulátor s n rotačními vazbami a n spoji obkreslující kružnici s úhlovým krokem α	53
Obr. 31: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/25$	55
Obr. 32: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/50$	55
Obr. 33: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/100$	56
Obr. 34: Čas výpočtu při průchodu singulární polohou, planární manipulátor $n = 2$, metoda Levenberg-Marquardt s koeficientem tlumení alfa	56
Obr. 35: Sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice ..	57
Obr. 36: Přesnost při sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice.....	57
Obr. 37: Čas výpočtu při sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice.....	57
Obr. 38: Singulární poloha osy A5 robota KUKA KR210 R2700 EXTRA (počáteční bod)	59
Obr. 39: Robot KUKA KR210 R2700 EXTRA obkreslující kružnici v rovině z a y.....	59
Obr. 40: Čas výpočtu jednoho úhlového kroku robota KUKA KR210 R2700 EXTRA bez průchodu singulární polohou	60
Obr. 41: Čas výpočtu jednoho úhlového kroku robota KUKA KR210 R2700 EXTRA s průchodem singulární polohou	60
Obr. 42: Sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu s úhlovým krokem $\pi/25$, inverze Jacobiho matice	61
Obr. 43: Přesnost jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu, úhlový krok $\pi/25$, inverze Jacobiho matice.....	61
Obr. 44: Čas výpočtu jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu, úhlový krok $\pi/25$	62

Obr. 45: Sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt	63
Obr. 46: Přesnost jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt	63
Obr. 47: Čas výpočtu jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt.....	63
Obr. 48: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením v singulární poloze, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (Pseudoinverze).....	64
Obr. 49: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením v singulární poloze, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (RLXA).....	64
Obr. 50: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením mimo singulární polohu, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (Pseudoinverze).....	64
Obr. 51: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením mimo singulární polohu, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (RLXA).....	64
Obr. 52: Zpětnovazební zapojení metody relaxace úhlu	66
Obr. 53: Metoda relaxace úhlu vyjádřená pomocí z-transformace	66
Obr. 54: Časový průběh minimalizace rezidua metodou relaxace úhlu a pseudoinverze, matice $n = 2$	67
Obr. 55: Časový průběh minimalizace rezidua metodou relaxace úhlu a pseudoinverze, matice $n = 10$	68
Obr. 56: Průběh ideální (červená) a předepsané (modrá) trajektorie.....	70
Obr. 57: Přesun bodu T_R k ideální trajektorii s omezením	71
Obr. 58: Relaxace délky	72
Obr. 59: Aproximační kinematický řetězec	73
Obr. 60: Optimální řešení metody relaxace délky.....	73
Obr. 61: Postupná relaxace aproximačního kinematického řetězce	74
Obr. 62: Výsledek metody relaxace délky (neoptimální)	75
Obr. 63: Výpočet pozice předposledního bodu P_2	75
Obr. 64: Volba počátečních podmínek	76
Obr. 65: Řešení inverzní kinematické úlohy dle počátečních podmínek.....	76
Obr. 66: Výpočet inverzní kinematické úlohy v robotice pomocí metody FABRIK	77

Obr. 67: Průchod kinematickým řetězcem pomocí metody FABRIK a metody relaxace délky.....	78
Obr. 68: Transformace sférické vazby na rotační pomocí paralelního virtuálního spoje	78

Seznam tabulek

Tab. 1: Porovnání konvergence vybraných numerických metod. 68: Transformace sférické vazby na rotační pomocí paralelního virtuálního spoje	31
Tab. 2: Časové porovnání vybraných numerických metod pro čtvercové matice.....	32
Tab. 3: Časové porovnání vybraných numerických metod pro obdélníkové matice.....	35
Tab. 4: Tabulka Denavit-Hartenbergových parametrů	45
Tab. 5: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice planárního manipulátoru z obrázku 30	54
Tab. 6: Denavit-Hartenbergovy parametry robota KUKA KR210 R2700 EXTRA.....	58
Tab. 7: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice průmyslového robota KUKA R210 R2700 EXTRA, poloměr kružnice 10 mm.....	60
Tab. 8: Časové porovnání vybraných heuristických metod.....	79

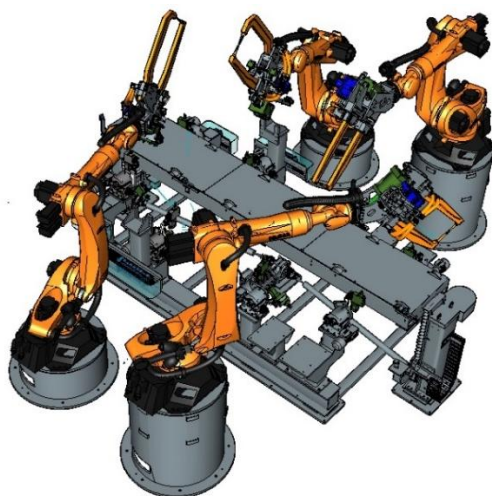
1 Úvod

Při zkoumání rozsáhlých robotických soustav pro potřeby firmy ŠKODA AUTO a.s. byl vytvořen software, který je zaměřen na kontrolu robotů a robotových standardů používaných v koncernu Volkswagen AG. Software pomáhá k rychlé orientaci v softwarovém prostředí svařovny a zároveň využívá matematických modelů pro stanovení neznámých proměnných při popisu chování robotů. Software nese název SKODA TOOL.

Tvorba softwaru SKODA TOOL má své opodstatnění. V současné době je v provozu ve svařovnách firmy ŠKODA AUTO a.s. přes dva tisíce průmyslových robotů. Tyto roboty obsahují přibližně osm set tisíc pohybových bodů, tzn. souřadnic popisujících jednotlivé trajektorie, a přibližně patnáct milionů řádků logických instrukcí. Dále je nutné poznamenat, že svařovna je dynamická oblast, kde dochází ke každodenním změnám, ať už v rámci nové výstavby, nebo v rámci udržování kvality stávající výroby. Jen v roce 2017 bylo nainstalováno v rámci svařoven firmy ŠKODA AUTO a.s. přes pět set nových robotů. Jakákoliv manuální správa dat je v tomto případě prakticky nemožná a neefektivní. To byl také důvod, který vedl k tvorbě softwaru SKODA TOOL, aby pomáhal specialistům při každodenní práci s roboty.

Software SKODA TOOL je zaměřen na kontrolu rozsáhlých robotových soustav. Příklad takové robotové soustavy je na obrázku 1. Podle koncernových předpisů může tvořit jednu soustavu až dvanáct robotů. Každá taková soustava robotů sdílí svůj operační prostor a čas. To znamená, že každá soustava robotů je specifická co do systému blokad, jednotlivých kroků operací a technologických nastavení. Každá soustava robotů se také generačně liší svým softwarovým a hardwarovým vybavením, jelikož se postupem času instalují novější generace robotů. Každý robot má svoje vlastní řízení a svoje vlastní programy. Je to nezávislá jednotka, která má na starosti konkrétní úkol, např. zavařit předepsané body na karoserii. To vše při spolupráci s ostatními roboty. Zároveň musí být roboty naprogramovány tak, aby nedošlo k jejich vzájemné srážce. Uvažujme běžnou situaci pro soustavu robotů. Robot nemůže vjet do společné kolizní oblasti s ostatními roboty, pokud se již v oblasti nachází jiný robot. Je proto nutné vhodně umístit začátek a konec blokace robota na trajektorii. Každá chyba při tvorbě programů může znamenat vážný problém. Pokud by programátor špatně naprogramoval tuto blokaci, hrozí zničení zařízení. To je pro výrobu v automobilovém průmyslu obrovské riziko. Před softwarem SKODA TOOL neexistoval robustní nástroj, který by podobné situace v rámci standardů Volkswagenu AG analyzoval a který by umožnil orientaci v tak rozsáhlém softwarovém prostředí svařovny. Před vznikem softwaru SKODA TOOL byla jediná možnost kontroly robotových standardů prostřednictvím aplikace Steineke. Steineke nabízel pouze omezené možnosti kontroly a výsledky byly v mnoha případech chybné, kvůli složité konfiguraci kontrolní rutiny. Nebyla zde možnost implementace vlastních kontrolních metod. Ty se proto nahrazovaly jednoúčelovými programy. Z tohoto důvodu se postupem času od aplikace

Steineke odstoupilo. Na základě zkušeností při tvorbě vlastních kontrolních metod, a na základě jasných potřeb firmy ŠKODA AUTO a.s., začal vznikat v roce 2013 software SKODA TOOL.



Obr. 1: Robotová soustava - dovářecí stanice

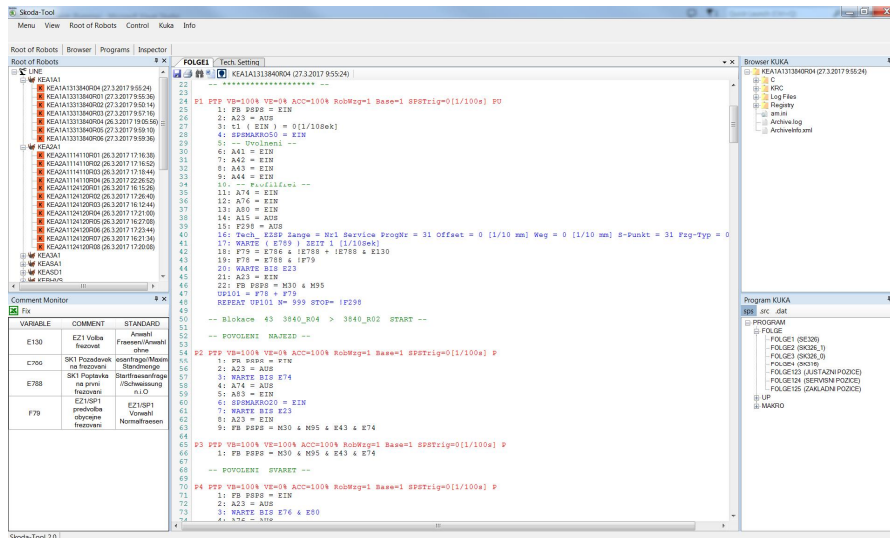
Při první analýze robotových programů softwarem SKODA TOOL na menším projektu se sto jedenácti roboty bylo zjištěno, že všechny roboty obsahovaly nějaké chyby. Tyto chyby jsou převážně způsobené lidským faktorem. Na základě přesné strojní kontroly bylo překontrolováno přes jeden a půl milionu robotových parametrů, a z toho bylo nalezeno přibližně třicet tři tisíc nedostatků. Jednalo se převážně o chyby v sintaxi programu. Dále o chyby, které mají vliv na kvalitu procesu. A dokonce byly nalezeny kritické chyby, které by mohly za určité situace způsobit kolizi robotů. Bez strojní kontroly by nikdy nebylo možné tak velký počet parametrů překontrolovat. Tímto způsobem software SKODA TOOL pomáhá snižovat prostojovost zařízení a zvyšovat kvalitu výroby.

Kromě kontroly programových chyb pomáhá software SKODA TOOL časově optimalizovat robotová pracoviště. Offline programy a trajektorie jsou v rámci výstavby ve svařovnách firmy ŠKODA AUTO a.s. modelovány v softwaru Process Simulace firmy Siemens. Často se stane, že programátoři nemají zkušenosti s tvorbou trajektorie a chováním reálného robota. Hotové programy tak nemusí být časově optimální. Z tohoto důvodu se prostřednictvím softwaru SKODA TOOL hledají časové potenciály na již zprovozněném zařízení.

Software SKODA TOOL je vytvořen v programovacím jazyce C#. Základním prvkem softwaru SKODA TOOL je prohlížeč, který umožňuje studovat strukturu robotových programů (viz obr. 2). Uživatel si může volit různé nástroje analýzy. Software obsahuje rutiny pro těžbu dat, které systematicky procházejí informace obsažené v robotových zálohách a formou srozumitelných přehledů informují uživatele o výsledcích, např. o použitém firmwaru, nebo svařovacích parametrech robotů. Další důležitou částí softwaru jsou kontrolní mechanismy postavené na parsování zdrojového kódu, které hlídají dodržování programovacích standardů koncernu Volkswagen AG.

Mezi základní části softwaru SKODA TOOL patří:

- prohlížeč robotových programů,
- kontrola robotových standardů,
- těžba dat a tvorba přehledů robotových parametrů,
- hledání časových potenciálů pro účely dráhové optimalizace,
- měření vybraných veličin reálného robota.



Obr. 2: Software SKODA TOOL

S postupem času se stal software SKODA TOOL platformou pro různé vědecké experimenty. Jak již bylo jednou řečeno, rozsáhlé robotové systavy skýtají potenciál pro nejrůznější dráhové optimalizace. Za tímto účelem bylo zapotřebí v rámci softwaru SKODA TOOL řešit několik základních inženýrských úloh. Jedná se především o přímou a inverzní kinematickou úlohu v robotice [1]. Dále pak řešení soustav lineárních rovnic o více proměnných [2-6]. Při zkoumání numerických metod v rámci programování uvedených úloh byly vyvinuty dva nové iterační algoritmy. Jedná se o metodu relaxace úhlu, která nám pomáhá při řešení soustav lineárních rovnic o více proměnných. Můžeme ji nepřímo použít i pro řešení inverzní kinematické úlohy v robotice. Druhý algoritmus vznikl jako obdoba metody relaxace úhlu. V tomto případě budeme hovořit o metodě relaxace délky, která je již přímo určena pro řešení inverzní kinematické úlohy v robotice.

Pro řešení inverzní kinematické úlohy v robotice bylo vyvinuto mnoho metod [11-29]. V rámci této disertační práce vybereme některé z nich a porovnáme je s námi objevenou metodou relaxace úhlu a relaxace délky. Větší část práce se budeme věnovat metodě relaxace úhlu. Ukážeme si její vlastnosti při výpočtu soustavy lineárních rovnic a následně provedeme experimenty s řešením inverzní kinematické úlohy v robotice. Metoda relaxace délky je vysloveně heuristickou metodou postavenou na jednoduchém geometrickém principu. Bude nás zajímat její chování ve srovnání s jinými heuristickými metodami řešení inverzní kinematické úlohy v robotice, jako je např. CCD (Cyclic

Coordinate Descent), nebo FABRIK (Forward And Backward Reaching Inverse Kinematics). Porovnání výpočetních vlastností odzkoušíme pomocí softwaru MATLAB. Výsledky teoretické části následně uplatníme v praktických úlohách v rámci softwaru SKODA TOOL.

Hlavní cíle disertační práce jsou:

- popsat metodu relaxace úhlu,
- ukázat vlastnosti metody relaxace úhlu (stabilita a rychlost konvergence),
- porovnat metodu relaxace úhlu s jinými numerickými metodami pro řešení soustav lineárních rovnic (Jacobiho metoda, Gauss-Seidelova metoda, metoda sdružených gradientů apod.),
- aplikovat metodu relaxace úhlu pro řešení inverzní kinematické úlohy v robotice,
- porovnat metodu relaxace úhlu a metodu relaxace délky s jinými algoritmy pro řešení inverzní kinematické úlohy v robotice (Newtonova metoda, metody postavené na Jacobiho matici, CCD apod.),
- ukázat postup dráhové optimalizace robota prostřednictvím softwaru SKODA TOOL a nové heuristické metody relaxace úhlu.

1.1 Zavedení matematických objektů

Metoda relaxace úhlu a metoda relaxace délky je ve své podstatě založena na geometrickém popisu bodů a vektorů v prostoru. Nadefinujeme si proto základní matematické objekty, které budeme v této práci používat [7-10].

Definice 1.1. Množinu $T \subset \mathbb{C}$, která je alespoň dvouprvková, nazveme **číselným tělesem** právě tehdy, když platí následující axiomy tělesa:

- 1) $(\forall \alpha \in T)(\forall \beta \in T)(\alpha + \beta \in T)$
- 2) $(\forall \alpha \in T)(\forall \beta \in T)(\alpha \cdot \beta \in T)$
- 3) $(\forall \alpha \in T)(-\alpha \in T)$
- 4) $(\forall \alpha \in T)(\alpha \neq 0)(1/\alpha \in T)$

Definice 1.2. Nechť je dáno číselné těleso T , neprázdná množina V , zobrazení $\oplus : V \times V \mapsto V$ a zobrazení $\odot : T \times V \mapsto V$. Prvky množiny V budeme nazývat vektory, zobrazení \oplus nazýváme sčítáním vektorů a zobrazení \odot násobením vektorů číslem z tělesa. Řekneme, že V je **vektorový prostor** nad tělesem T s vektorovými operacemi \oplus a \odot , tzn. $\{V, T, \oplus, \odot\}$, právě když platí následující axiomy vektorového prostoru:

- 1) $(\forall \mathbf{x} \in V)(\forall \mathbf{y} \in V)(\mathbf{x} \oplus \mathbf{y} = \mathbf{y} \oplus \mathbf{x})$
- 2) $(\forall \mathbf{x} \in V)(\forall \mathbf{y} \in V)(\forall \mathbf{z} \in V)(\mathbf{x} \oplus (\mathbf{y} \oplus \mathbf{z}) = (\mathbf{x} \oplus \mathbf{y}) \oplus \mathbf{z})$
- 3) $(\exists \mathbf{o} \in V)(\forall \mathbf{x} \in V)(\mathbf{x} \oplus \mathbf{o} = \mathbf{x})$
- 4) $(\exists \mathbf{x} \in V)(\forall \mathbf{y} \in V)(\mathbf{x} \oplus \mathbf{y} = \mathbf{o})$
- 5) $(\forall \alpha \in T)(\forall \beta \in T)(\forall \mathbf{x} \in V)(\alpha \odot (\beta \odot \mathbf{x}) = (\alpha \cdot \beta) \odot \mathbf{x})$
- 6) $(\forall \mathbf{x} \in V)(1 \odot \mathbf{x} = \mathbf{x})$
- 7) $(\forall \alpha \in T)(\forall \beta \in T)(\forall \mathbf{x} \in V)((\alpha + \beta) \odot \mathbf{x} = (\alpha \odot \mathbf{x}) \oplus (\beta \odot \mathbf{x}))$
- 8) $(\forall \alpha \in T)(\forall \mathbf{x} \in V)(\forall \mathbf{y} \in V)((\alpha \odot (\mathbf{x} \oplus \mathbf{y})) = (\alpha \odot \mathbf{x}) \oplus (\alpha \odot \mathbf{y}))$

Úmluva Jelikož budeme v této práci sčítání vektorů a násobením vektorů číslem z tělesa standardně používat, nahradíme symboly \oplus a \odot , symbolem $\mathbf{x} + \mathbf{y}$ pro sčítání vektorů a symbolem $\alpha \cdot \mathbf{x}$ pro násobením vektorů číslem z tělesa.

Definice 1.3. Nechť je V vektorový prostor nad číselným tělesem T . Zobrazení $\langle \cdot | \cdot \rangle : V \times V \rightarrow T$ nazveme **skalárním součinem**, jestliže platí:

- 1) $\langle \mathbf{x} | \mathbf{y} \rangle = \overline{\langle \mathbf{y} | \mathbf{x} \rangle}$
- 2) $\langle \mathbf{x} + \mathbf{y} | \mathbf{z} \rangle = \langle \mathbf{x} | \mathbf{z} \rangle + \langle \mathbf{y} | \mathbf{z} \rangle$
- 3) $\langle \alpha \mathbf{x} | \mathbf{y} \rangle = \alpha \langle \mathbf{x} | \mathbf{y} \rangle$
- 4) $\langle \mathbf{x} | \mathbf{x} \rangle \geq 0$
- 5) $\langle \mathbf{x} | \mathbf{x} \rangle = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$

Definice 1.4. Necht' je V vektorový prostor nad číselným tělesem T . Zobrazení $\|\cdot\|: V \rightarrow T$ se nazve **normou**, jestliže platí:

- 1) $\|\mathbf{x}\| \geq 0$
- 2) $\|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$
- 3) $\|\alpha\mathbf{x}\| = |\alpha|\|\mathbf{x}\|$
- 4) $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$

Vektorový prostor V , na kterém je definovaná norma $\|\cdot\|$, se nazývá **normovaný prostor** $\{V, \|\cdot\|\}$.

Definice 1.5. Necht' je M množina. Funkci $\rho: M \times M \rightarrow T$ nazveme **metrikou**, jestliže splňuje následující podmínky:

- 1) pro každé $x \in M$ je $\rho(x, x) = 0$
- 2) pro každé $x, y \in M, x \neq y$ je $\rho(x, y) = \rho(y, x) > 0$
- 3) pro každé $x, y, z \in M$ platí $\rho(x, z) \leq \rho(x, y) + \rho(y, z)$

Množinu M , na které je definovaná metrika ρ , nazýváme **metrický prostor** $\{M, \rho\}$.

Definice 1.6. Metrický prostor $\{M, \rho\}$ je **úplný**, jestliže každá cauchyovská posloupnost prvků je v něm konvergentní.

Definice 1.7. V metrickém prostoru $\{M, \rho\}$ nazveme **posloupnost cauchyovskou**, pokud platí $\forall \varepsilon > 0 \exists n_0 \in \mathbb{N} \forall m, n \geq n_0: \rho(x_m, x_n) < \varepsilon$.

Definice 1.8. Necht' V je vektorový prostor, na němž je zadán libovolný skalární součin $\langle \cdot | \cdot \rangle$. Necht' norma $\|\cdot\|$ je generována skalárním součinem a $\rho(\cdot, \cdot)$ je metrika generovaná touto normou. Necht' metrický prostor $\{V, \rho\}$ je úplný. Pak čtveřici $\{V, \langle \cdot | \cdot \rangle, \|\cdot\|, \rho(\cdot, \cdot)\}$ budeme nazývat **Hilbertovým prostorem** a značit \mathcal{H} .

Úmluva Pro účely této práce budeme uvažovat Hilbertův prostor, tzn. čtveřici $\{\mathbb{R}^n, \langle \cdot | \cdot \rangle_s, \|\cdot\|_e, \rho_e(\cdot, \cdot)\}$, kde vektorový prostor V je nad tělesem \mathbb{R} , $\langle \cdot | \cdot \rangle_s$ je standardní skalární součin, $\|\cdot\|_e$ je euklidovská norma generovaná standardním skalárním součinem a $\rho_e(\cdot, \cdot)$ je euklidovská metrika generovaná euklidovskou normou $\|\cdot\|_e$. V konečné instanci budeme pracovat pouze s konečnou dimenzí \mathbb{R} , čímž můžeme nazvat čtveřici $\{\mathbb{R}^n, \langle \cdot | \cdot \rangle_s, \|\cdot\|_e, \rho_e(\cdot, \cdot)\}$ euklidovským prostorem \mathbb{E}_n .

Definice 1.9. Standardní skalární součin $\langle \cdot | \cdot \rangle_s$ libovolných dvou reálných vektorů $\mathbf{x}, \mathbf{y} \in \mathbb{E}_n$ definujeme jako reálné číslo:

$$\langle \mathbf{x} | \mathbf{y} \rangle_s = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^n x_i y_i \quad (1)$$

Definice 1.10. Euklidovskou normu $\| \cdot \|_e$ generovanou standardním skalárním součinem $\langle \cdot | \cdot \rangle_s$ libovolného reálného vektoru $\mathbf{x} \in \mathbb{E}_n$ definujeme jako reálné číslo:

$$\| \mathbf{x} \|_e = \sqrt{\langle \mathbf{x} | \mathbf{x} \rangle_s} = \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{\sum_{i=1}^n x_i^2} \quad (2)$$

Pozn.: Euklidovská norma nám v geometrickém pojetí prostoru vyjadřuje velikost vektoru \mathbf{x} .

Definice 1.11. Euklidovskou metriku $\rho_e(\cdot, \cdot)$ generovanou euklidovskou normou $\| \cdot \|_e$ libovolných dvou bodů $P_1, P_2 \in \mathbb{E}_n$ definujeme jako reálné číslo:

$$\begin{aligned} \rho_e(P_1, P_2) &= \| P_1 - P_2 \|_e = \sqrt{\langle (P_1 - P_2) | (P_1 - P_2) \rangle_s} \\ &= \sqrt{(P_1 - P_2)^T (P_1 - P_2)} = \sqrt{\sum_{i=1}^n (P_{1i} - P_{2i})^2} \end{aligned} \quad (3)$$

Pozn.: Euklidovská metrika nám v geometrickém pojetí vyjadřuje vzdálenost dvou bodů v prostoru \mathbb{E}_n .

Úmluva Pokud budeme hovořit v aplikaci o bodech v prostoru \mathbb{E}_n , myslíme tím jejich polohový vektor $\mathbf{p} = \overrightarrow{OP}$, který nám jednoznačně určuje bod P . Zápis P a \mathbf{p} budeme proto považovat za rovnocenný. Přičemž souřadnice bodů budeme z formálního hlediska zapisovat $[x_1, x_2, x_3, \dots, x_n]$ a souřadnice vektorů $(x_1, x_2, x_3, \dots, x_n)$. Uvažujme standardní bázi $\langle \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \dots, \mathbf{e}_n \rangle$ prostoru \mathbb{E}_n , pak každý vektor je určený lineární kombinací této báze (4)

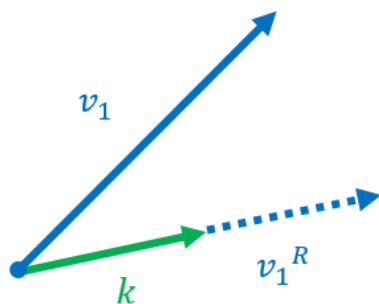
$$\mathbf{x} = x_1 \mathbf{e}_1 + x_2 \mathbf{e}_2 + \dots + x_n \mathbf{e}_n = \sum_{i=1}^n x_i \mathbf{e}_i \quad (4)$$

Nyní máme k dispozici matematický aparát, který nás bude provázet celou prací. Můžeme konstatovat, že výpočet euklidovské normy je základním kamenem nové heuristické metody relaxace úhlu a metody relaxace délky. Obě metody vyžadují pouze základní matematické operace při práci s vektory, a z tohoto důvodu je jejich implementace velice snadná.

2 Metoda relaxace úhlu

2.1 Popis metody relaxace úhlu

Metoda relaxace úhlu je určena pro řešení soustav lineárních rovnic více proměnných. Jak již název metody napovídá, základním kamenem algoritmu bude operace, kterou nazveme relaxace úhlu. Na následujícím příkladu si vysvětlíme, co tato operace znamená. Uvažujme libovolný vektor \mathbf{v}_1 a vektor \mathbf{k} (viz obr. 3). Vektor \mathbf{k} nám ukazuje směr, do kterého chceme vektor \mathbf{v}_1 otočit. Vznikne nám tak nový vektor \mathbf{v}_1^R , který je lineárně závislý s vektorem \mathbf{k} a má velikost vektoru \mathbf{v}_1 .



Obr. 3: Relaxace úhlu

Definice 2.1. Relaxaci úhlu definujeme jako otočení vektoru \mathbf{v}_1 do směru \mathbf{k} (5). Tento jednoduchý výpočet bude základním stavebním kamenem celého algoritmu.

$$\mathbf{v}_1^R = \frac{\mathbf{k}}{\|\mathbf{k}\|_e} \|\mathbf{v}_1\|_e \quad (5)$$

Nyní si odvodíme celý postup pro řešení soustav lineárních rovnic pomocí metody relaxace úhlu. Začneme s definicí soustavy lineárních rovnic.

Definice 2.2. Soustavou m lineárních rovnic o n neznámých $x_1, x_2, x_3, \dots, x_n$ nazveme soustavu (6) kde a_{11}, \dots, a_{mn} a $b_1, \dots, b_m \in \mathbb{R}$.

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n &= b_m \end{aligned} \quad (6)$$

Řešením soustavy (6) se nazývá každá uspořádaná n -tice $(x_1, x_2, x_3, \dots, x_n)^T \in \mathbb{R}$, tj. n -členný vektor, který splňuje všechny rovnice soustavy (6).

Soustavu lineárních rovnic lze zapsat v maticovém tvaru $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, kde \mathbf{A} je matice soustavy, \mathbf{b} je vektor pravých stran a \mathbf{x} je vektor neznámých.

$$\mathbf{A} = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}$$

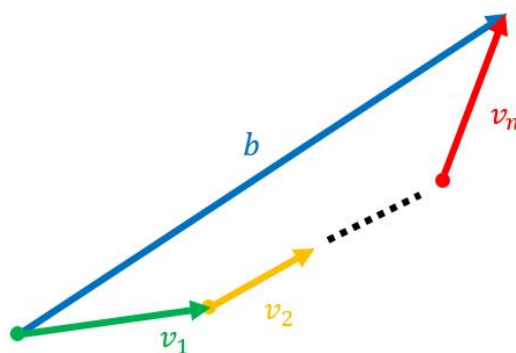
V našem případě budeme vycházet z ještě jiné možné podoby zápisu. Na soustavu budeme nahlížet z geometrického hlediska. Vektor pravých stran \mathbf{b} je roven lineární kombinaci sloupců matice soustavy \mathbf{A} (7).

$$\begin{aligned} \mathbf{A}_1 \cdot x_1 + \mathbf{A}_2 \cdot x_2 + \cdots + \mathbf{A}_n \cdot x_n &= \mathbf{b} \\ \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix} \cdot x_1 + \begin{pmatrix} a_{12} \\ \vdots \\ a_{m2} \end{pmatrix} \cdot x_2 + \cdots + \begin{pmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{pmatrix} \cdot x_n &= \mathbf{b} \end{aligned} \quad (7)$$

Tento zápis můžeme přepsat jako součet vektorů \mathbf{v}_1 až \mathbf{v}_n (8). Vektory \mathbf{v}_1 až \mathbf{v}_n jsou lineárně závislé se sloupci matice soustavy \mathbf{A} .

$$\begin{aligned} \mathbf{v}_1 + \mathbf{v}_2 + \cdots + \mathbf{v}_n &= \mathbf{b} \\ \mathbf{v}_1 = \mathbf{A}_1 \cdot x_1, \mathbf{v}_2 = \mathbf{A}_2 \cdot x_2, \dots, \mathbf{v}_n = \mathbf{A}_n \cdot x_n \end{aligned} \quad (8)$$

Nyní můžeme soustavu lineárních rovnic vyjádřit z geometrického hlediska také graficky. Pro ilustraci budeme uvažovat pouze euklidovský prostor \mathbb{E}_2 , případně \mathbb{E}_3 . Pro vyšší dimenze již grafické znázornění vektorů v euklidovském prostoru není interpretovatelné. Obecně je však možné metodu relaxace úhlu pro řešení soustav lineárních rovnic zobecnit pro \mathbb{E}_n , přičemž platí stále stejné principy. Na obrázku 4. vidíme geometrickou interpretaci soustavy lineárních rovnic pomocí vektorů.



Obr. 4: Geometrická interpretace soustavy lineárních rovnic pomocí vektorů

Obecně můžeme vektor pravých stran \mathbf{b} sestavit z libovolného počtu vektorů, které ovšem nemusí odpovídat soustavě lineárních rovnic, tzn. vektory nebudou lineárně závislé se sloupci matice soustavy \mathbf{A} . Zavedeme si proto dva nové pojmy, které budou specifické pro tuto práci a obecně se nepoužívají, nicméně nám pomohou k velmi názorné interpretaci algoritmu.

Definice 2.3. Nevlastním řešením soustavy budeme nazývat vektory \mathbf{v}_1 až \mathbf{v}_n , které jsou v součtu rovny vektoru pravých stran \mathbf{b} , ale nemusí být lineárně závislé se sloupci matice soustavy \mathbf{A} . Přičemž $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n$ jsou diagonální matice, kde diagonální složky nejsou stejně velké (9).

$$\mathbf{v}_1 + \mathbf{v}_2 + \dots + \mathbf{v}_n = \mathbf{b}$$

$$\mathbf{X}_i = \begin{pmatrix} x_{i1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & x_{im} \end{pmatrix}_i \quad (9)$$

$$\mathbf{v}_1 = \mathbf{X}_1 \cdot \mathbf{A}_1, \mathbf{v}_2 = \mathbf{X}_2 \cdot \mathbf{A}_2, \dots, \mathbf{v}_n = \mathbf{X}_n \cdot \mathbf{A}_n$$

Definice 2.4. Vlastním řešením soustavy budeme nazývat vektory, které jsou v součtu rovny vektoru pravých stran \mathbf{b} a zároveň jsou lineárně závislé se sloupci matice soustavy \mathbf{A} . Přičemž x_1, x_2, \dots, x_n jsou skaláry (10).

$$\mathbf{v}_1 + \mathbf{v}_2 + \dots + \mathbf{v}_n = \mathbf{b}$$

$$\mathbf{v}_1 = \mathbf{A}_1 \cdot x_1, \mathbf{v}_2 = \mathbf{A}_2 \cdot x_2, \dots, \mathbf{v}_n = \mathbf{A}_n \cdot x_n \quad (10)$$

Nalézt vlastní řešení soustavy je náš cíl, ale zároveň určit, nebo odhadnout x_1, x_2, \dots, x_n je složité. Budeme proto postupovat tak, že nejprve nalezneme nevlastní řešení soustavy. To můžeme udělat velice jednoduše tak, že rozdělíme vektor \mathbf{b} na několik stejných částí podle počtu hledaných neznámých n . Je jasné, že \mathbf{v}_{10} až \mathbf{v}_{n0} nemusí být lineárně závislé se sloupci matice soustavy \mathbf{A}_1 až \mathbf{A}_n (11).

$$\mathbf{v}_{10} + \mathbf{v}_{20} + \dots + \mathbf{v}_{n0} = \mathbf{b}$$

$$\mathbf{v}_{10} = \frac{\mathbf{b}}{n}, \mathbf{v}_{20} = \frac{\mathbf{b}}{n}, \dots, \mathbf{v}_{n0} = \frac{\mathbf{b}}{n} \quad (11)$$

Následně musíme převést nevlastní řešení soustavy na vlastní řešení soustavy. Právě k tomuto převodu použijeme relaxaci úhlu (5) z definice 2.1. Tento krok je esenciálním prvkem celé metody. Z lineárně nezávislých vektorů \mathbf{v}_{10} až \mathbf{v}_{n0} učiníme vektory lineárně závislé k \mathbf{A}_1 až \mathbf{A}_n i za cenu toho, že nedosáhneme přesného řešení soustavy a zbyde nám odchylka, kterou nazveme reziduum \mathbf{r}_k (12). Relaxaci úhlu se převede matice \mathbf{X}_i do skalární podoby x_i .

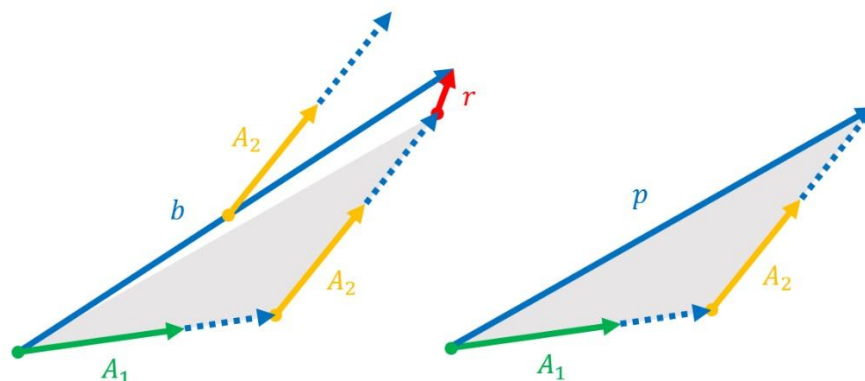
$$\pm \frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} \|\mathbf{v}_{10}\|_e \pm \frac{\mathbf{A}_2}{\|\mathbf{A}_2\|_e} \|\mathbf{v}_{20}\|_e \pm \dots \pm \frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} \|\mathbf{v}_{n0}\|_e + \mathbf{r}_0 = \mathbf{b}$$

$$\pm \mathbf{v}_{10}^R \pm \mathbf{v}_{20}^R \pm \dots \pm \mathbf{v}_{n0}^R + \mathbf{r}_0 = \mathbf{b}$$

$$\mathbf{p}_0 = \sum_{i=1}^n \pm \mathbf{v}_{i0}^R$$

$$\mathbf{p}_0 + \mathbf{r}_0 = \mathbf{b} \quad (12)$$

Příklad převodu pro $n = 2$ z nevlastního řešení soustavy na vlastní řešení soustavy je graficky znázorněn na obrázku 5.



Obr. 5: Geometrická interpretace převodu nevlastního řešení soustavy na řešení soustavy vlastní

Obecně neplatí, že se vektory musí pro co nejmenší reziduum \mathbf{r}_k sčítat. Abychom získali co nejmenší reziduum, musíme při každé relaxaci vyhodnotit správné znaménko (v zápisu je obecně ponecháno \pm viz rovnice 12). Do výpočtu nám totiž vstupuje pouze délka vektoru $\|\mathbf{v}_{nk}\|_e$, čímž ztrácíme informaci o orientaci vektoru \mathbf{b} . Pokud by vektor \mathbf{b} byl záporný a orientace některého z vektorů \mathbf{A}_1 až \mathbf{A}_n kladná, reziduum by se při sčítání zvětšovalo. Algoritmus by pak divergoval. Rozhodnout o znaménku můžeme tak, že porovnáme, která varianta se více přiblíží k vektoru \mathbf{b} .

Pokud bude znaménko kladné, platí nerovnice 13.

$$\left\| \frac{\mathbf{b}}{n} - \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e < \left\| \frac{\mathbf{b}}{n} + \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e \quad (13)$$

Pokud bude znaménko záporné, platí nerovnice 14.

$$\left\| \frac{\mathbf{b}}{n} - \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e > \left\| \frac{\mathbf{b}}{n} + \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e \quad (14)$$

Rovnost by odpovídala pouze nulovému vektoru \mathbf{b} .

Tento zápis se dá vyjádřit pomocí znaménkové funkce signum (15)

$$\text{sgn} \left(\left\| \frac{\mathbf{b}}{n} + \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e - \left\| \frac{\mathbf{b}}{n} - \frac{\mathbf{A}_i}{\|\mathbf{A}_i\|_e} \|\mathbf{v}_{i0}\|_e \right\|_e \right)_n \quad (15)$$

V druhém kroku algoritmu prohlásíme reziduum \mathbf{r}_0 novým hledaným vektorem pravých stran $\mathbf{b}_0 = \mathbf{r}_0$ a provedeme identické operace. Nejprve nalezneme nevlastní řešení soustavy (16).

$$\begin{aligned} \mathbf{v}_{11} + \mathbf{v}_{21} + \dots + \mathbf{v}_{n1} &= \mathbf{b}_0 \\ \mathbf{v}_{11} &= \frac{\mathbf{b}_0}{n}, \mathbf{v}_{21} = \frac{\mathbf{b}_0}{n}, \dots, \mathbf{v}_{n1} = \frac{\mathbf{b}_0}{n} \end{aligned} \quad (16)$$

Následně převedeme vektory na vlastní řešení soustavy pomocí relaxace úhlu (17).

$$\begin{aligned}
\pm \frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} \|\mathbf{v}_{11}\|_e \pm \frac{\mathbf{A}_2}{\|\mathbf{A}_2\|_e} \|\mathbf{v}_{21}\|_e \pm \dots \pm \frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} \|\mathbf{v}_{n1}\|_e + \mathbf{r}_1 = \mathbf{b}_0 \\
\pm \mathbf{v}_{11}^R \pm \mathbf{v}_{21}^R \pm \dots \pm \mathbf{v}_{n1}^R + \mathbf{r}_1 = \mathbf{b}_0 \\
\mathbf{p}_1 = \sum_{i=1}^n \pm \mathbf{v}_{i1}^R \\
\mathbf{p}_1 + \mathbf{r}_1 = \mathbf{b}_0 = \mathbf{r}_0
\end{aligned} \tag{17}$$

V každém kroku algoritmu se velikost rezidua $\|\mathbf{r}_k\|_e$ zmenšuje, až se začne infinitesimalně blížit nulovému vektoru \mathbf{o} . Tento předpoklad si později dokážeme. Celkově pak platí, že součet všech vektorů \mathbf{p}_i a posledního rezidua \mathbf{r}_k je roven vektoru pravých stran \mathbf{b} (18).

$$\mathbf{b} = \sum_{i=0}^k \mathbf{p}_i + \mathbf{b}_k = \sum_{i=0}^k \mathbf{p}_i + \mathbf{r}_k \tag{18}$$

Abychom mohli vyjádřit hledané neznámé x_1, x_2 až x_n , musíme sečíst lineárně závislé vektory k jednotlivým sloupcům matice soustavy \mathbf{A}_1 až \mathbf{A}_n z každého kroku iterace. Soustavu rovnic 19 převedeme na rovnici 20. Vektory \mathbf{v}_{1c} až \mathbf{v}_{nc} jsou lineárně závislé ke sloupcům matice soustavy \mathbf{A}_1 až \mathbf{A}_n .

$$\begin{aligned}
\pm \mathbf{v}_{10}^R \pm \mathbf{v}_{20}^R \pm \dots \pm \mathbf{v}_{n0}^R = \mathbf{p}_0 \\
\pm \mathbf{v}_{11}^R \pm \mathbf{v}_{21}^R \pm \dots \pm \mathbf{v}_{n1}^R = \mathbf{p}_1 \\
\vdots
\end{aligned} \tag{19}$$

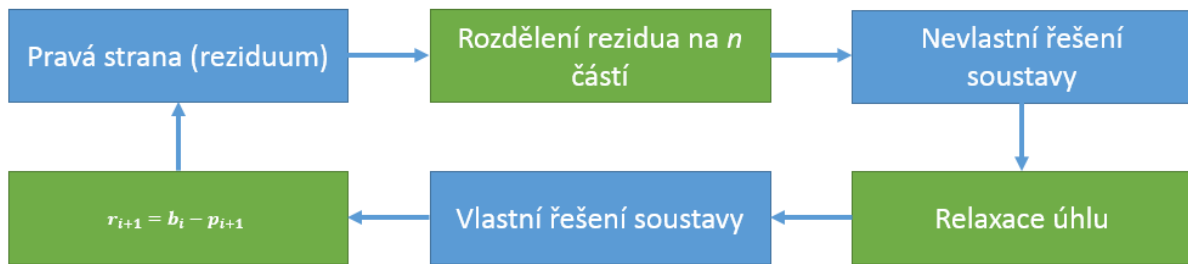
$$\begin{aligned}
\pm \mathbf{v}_{1k}^R \pm \mathbf{v}_{2k}^R \pm \dots \pm \mathbf{v}_{nk}^R = \mathbf{p}_k \\
\sum_{i=0}^k \pm \mathbf{v}_{1i}^R + \sum_{i=0}^k \pm \mathbf{v}_{2i}^R + \dots + \sum_{i=0}^k \pm \mathbf{v}_{ni}^R = \sum_{i=0}^k \mathbf{p}_i = \mathbf{b} - \mathbf{r}_k
\end{aligned} \tag{20}$$

$$\mathbf{v}_{1c} + \mathbf{v}_{1c} + \dots + \mathbf{v}_{nc} = \mathbf{p}_c = \mathbf{b} - \mathbf{r}_k$$

Posledním krokem je vyjádření hledaných neznámých x_1, x_2 až x_n na základě rovnice 10 a 20. Jelikož x_1, x_2 až x_n je po relaxaci úhlu skalární hodnota, stačí vydělit libovolný prvek lineárně závislého vektoru \mathbf{v}_{1c} až \mathbf{v}_{nc} odpovídajícím prvkem z vektoru \mathbf{A}_1 až \mathbf{A}_n (21). Musíme si dát pozor na dělení nulou a pro výpočet vzít nenulový prvek z vektoru \mathbf{A}_1 až \mathbf{A}_n . Matice soustavy \mathbf{A} může totiž obsahovat nulové prvky. Přesnost řešení odpovídá velikosti rezidua \mathbf{r}_k . Pokud je $\mathbf{r}_k = \mathbf{o}$, dosáhli jsme přesného řešení, jelikož $\mathbf{p}_c = \mathbf{b} - \mathbf{r}_k$.

$$x_1 = \frac{\mathbf{v}_{1cm}}{\mathbf{A}_{1m}}, x_2 = \frac{\mathbf{v}_{2cm}}{\mathbf{A}_{2m}}, \dots, x_n = \frac{\mathbf{v}_{ncm}}{\mathbf{A}_{nm}} \tag{21}$$

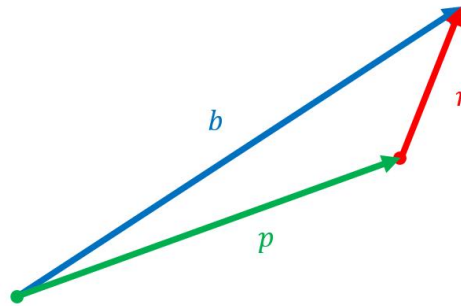
Pro názornost vyjádříme algoritmus blokově pomocí základních operací viz obrázek 6.



Obr. 6: Základní operace algoritmu relaxace úhlu

2.2 Konvergence a stabilita metody relaxace úhlu

Důkaz konvergence metody relaxace úhlu je založený na myšlence, že velikost rezidua $\|\mathbf{r}\|_e$ se v každém kroku metody zmenšuje. Budeme vycházet z trojúhelníkové nerovnosti (obr. 7)(22).



Obr. 7: Trojúhelníková nerovnost

$$\begin{aligned}
 \|\mathbf{p} + \mathbf{r}\|_e &\leq \|\mathbf{p}\|_e + \|\mathbf{r}\|_e \\
 \|\mathbf{b}\|_e &\leq \|\mathbf{p}\|_e + \|\mathbf{r}\|_e \\
 \|\mathbf{b}\|_e - \|\mathbf{p}\|_e &\leq \|\mathbf{r}\|_e
 \end{aligned} \tag{22}$$

Za vektor \mathbf{p} dosadíme vztah pro relaxaci úhlu (23)

$$\|\mathbf{b}\|_e - \left\| \pm \frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} \|\mathbf{v}_1\|_e \pm \frac{\mathbf{A}_2}{\|\mathbf{A}_2\|_e} \|\mathbf{v}_2\|_e \pm \dots \pm \frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} \|\mathbf{v}_n\|_e \right\| \leq \|\mathbf{r}\|_e \tag{23}$$

Pro nevlastní řešení soustavy jsme volili \mathbf{v}_1 až $\mathbf{v}_n = \frac{\mathbf{b}}{n}$ (24). Obecně lze volit i jiné možné rozdělení vektoru \mathbf{b} na n různých částí. Pozn.: volba rozdělení má vliv na rychlost konvergence a stabilitu řešení metody. Pro $\frac{\mathbf{b}}{n}$ bude metoda konvergovat vždy.

$$\|\mathbf{b}\|_e - \left\| \pm \frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} \left\| \frac{\mathbf{b}}{n} \right\|_e \pm \frac{\mathbf{A}_2}{\|\mathbf{A}_2\|_e} \left\| \frac{\mathbf{b}}{n} \right\|_e \pm \dots \pm \frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} \left\| \frac{\mathbf{b}}{n} \right\|_e \right\| \leq \|\mathbf{r}\|_e \tag{24}$$

Následně celou nerovnici upravíme a vytkneme konstanty (25).

$$\begin{aligned} \|\mathbf{b}\|_e - \left\| \frac{\mathbf{b}}{n} \right\|_e \cdot \left\| \pm \frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} \pm \frac{\mathbf{A}_2}{\|\mathbf{A}_2\|_e} \pm \dots \pm \frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} \right\| &\leq \|\mathbf{r}\|_e \\ \|\mathbf{b}\|_e - \frac{1}{n} \|\mathbf{b}\|_e \cdot \left\| \pm \widehat{\mathbf{A}}_1 \pm \widehat{\mathbf{A}}_2 \pm \dots \pm \widehat{\mathbf{A}}_n \right\| &\leq \|\mathbf{r}\|_e \end{aligned} \quad (25)$$

Jelikož vektory $\frac{\mathbf{A}_1}{\|\mathbf{A}_1\|_e} = \widehat{\mathbf{A}}_1$ až $\frac{\mathbf{A}_n}{\|\mathbf{A}_n\|_e} = \widehat{\mathbf{A}}_n$ jsou normované, může velikost jednoho z prvků $\widehat{\mathbf{A}}_1$ až $\widehat{\mathbf{A}}_n$ dosáhnout maximální hodnoty ± 1 , např. $\widehat{\mathbf{A}}_1 = (1,0)^T$, $\widehat{\mathbf{A}}_2 = (-1,0)^T$, $\widehat{\mathbf{A}}_3 = (0,-1)^T$ nebo $\widehat{\mathbf{A}}_i = (0,1)^T$. Z tohoto důvodu se celková velikost součtu prvků na pozici j normovaných vektorů $\widehat{\mathbf{A}}_1$ až $\widehat{\mathbf{A}}_n$ bude pohybovat v maximálním intervalu $\langle 0, \pm n \rangle_j$. První extrém odpovídá situaci, kdy se vektory vzájemně odečtou. Druhý extrém odpovídá situaci, kdy existuje vždy jeden prvek vektoru s maximální hodnotou ± 1 , zbylé prvky musí být nulové. Pokud je pozice j maximálního prvku ve vektoru $\widehat{\mathbf{A}}_1$ až $\widehat{\mathbf{A}}_n$ stejná, můžeme dosáhnout maximálního součtu $\pm n$ (26).

$$\pm \widehat{\mathbf{A}}_{1j} \pm \widehat{\mathbf{A}}_{2j} \pm \dots \pm \widehat{\mathbf{A}}_{nj} \in \langle 0, \pm n \rangle_j \quad j = 1, 2, \dots, m \quad (26)$$

Celou nerovnici vyjádříme zjednodušeným vztahem 27. Přičemž výstupem z normy $\|(\langle 0, \pm n \rangle_1, \langle 0, \pm n \rangle_2, \dots, \langle 0, \pm n \rangle_m)^T\|_e$ musí být opět číslo z intervalu $\langle 0, n \rangle$.

$$\|\mathbf{b}\|_e - \frac{1}{n} \|\mathbf{b}\|_e \cdot \|(\langle 0, \pm n \rangle_1, \langle 0, \pm n \rangle_2, \dots, \langle 0, \pm n \rangle_m)^T\|_e \leq \|\mathbf{r}\|_e \quad (27)$$

Řešení se bude pohybovat mezi dvěma extrémy. První extrém odpovídá vzájemnému odečtení vektorů. Potom z vektoru \mathbf{b} žádný vektor neodečteme a reziduum \mathbf{r} bude odpovídat vektoru \mathbf{b} (28).

$$\begin{aligned} \|\mathbf{b}\|_e - \frac{1}{n} \|\mathbf{b}\|_e \cdot \|\mathbf{o}\|_e &\leq \|\mathbf{r}\|_e \\ \|\mathbf{b}\|_e &= \|\mathbf{r}\|_e \end{aligned} \quad (28)$$

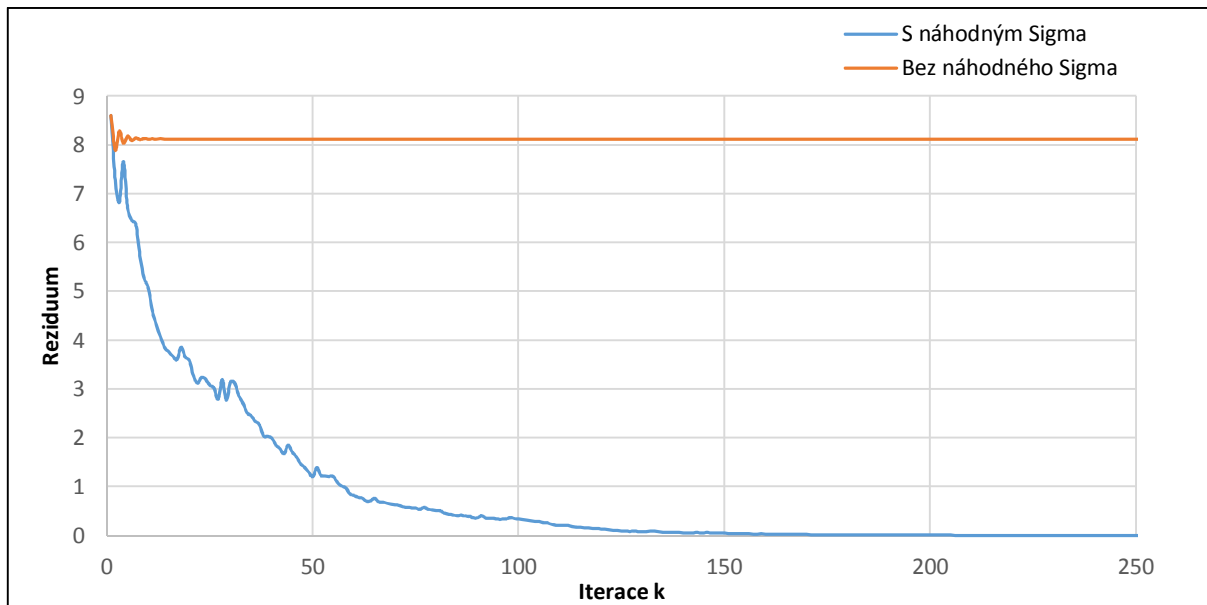
Druhý extrém odpovídá situaci, kdy dosáhneme maximálního součtu. Například v případě prvního prvku je velikost normy $\|(\pm n, 0, \dots, 0)^T\|_e = n$. Reziduum \mathbf{r} pak bude větší, nebo rovno nule (29).

$$\begin{aligned} \|\mathbf{b}\|_e - \frac{1}{n} \|\mathbf{b}\|_e \cdot \|(\pm n, 0, \dots, 0)^T\|_e &\leq \|\mathbf{r}\|_e \\ \|\mathbf{b}\|_e - \frac{1}{n} \|\mathbf{b}\|_e \cdot n &\leq \|\mathbf{r}\|_e \\ 0 &\leq \|\mathbf{r}\|_e \end{aligned} \quad (29)$$

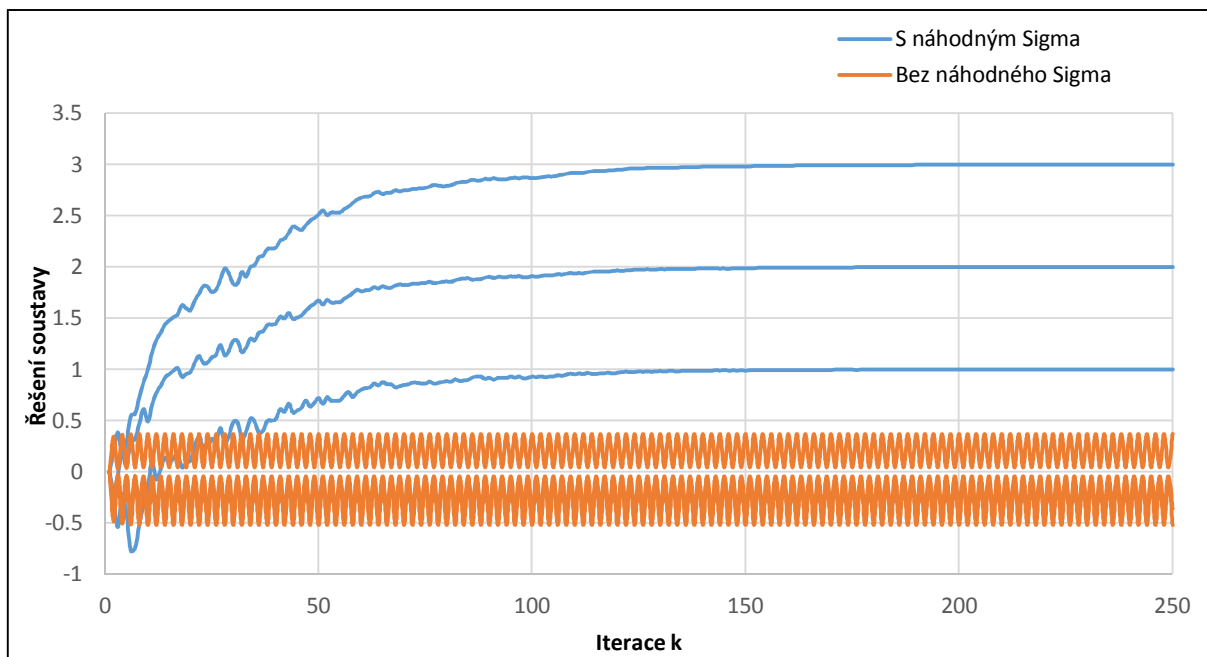
Nyní můžeme konstatovat, že velikost rezidua $\|\mathbf{r}\|_e$ se bude vždy pohybovat v intervalu $\langle 0, \|\mathbf{b}\|_e \rangle$, jelikož dle nerovnice 28 a 29 se musí pohybovat mezi těmito extrémy. To znamená, že se $\|\mathbf{r}\|_e$ musí v každém kroku metody zmenšovat až na mezní situaci $\|\mathbf{r}\|_e = \|\mathbf{b}\|_e$. Je splněno tzv. D'Alembertovo kritérium konvergence. Mezní situace $\|\mathbf{r}\|_e = \|\mathbf{b}\|_e$ může zapříčinit divergenci,

stagnování a oscilaci algoritmu. D'Alembertovo kritérium zaručuje konvergenci, ale není zárukou limitní konvergence k nule. Algoritmus by mohl konvergovat k jiné hodnotě. Zavedeme si proto do metody proměnnou σ , která bude nabývat náhodných hodnot v intervalu (0,1). Při volbě nevlastního řešení soustavy nebudeme dělit celý vektor \mathbf{b} , resp. $\mathbf{b}_0, \mathbf{b}_1$, atd. na stejné části, ale pouze jeho náhodnou část z intervalu (0,1) (30). Tento krok nám odstraní problémy pro mezní situace a také zaručí konvergenci algoritmu k nule. Zavedení náhodného prvku může také zlepšit rychlost konvergence (viz obr. 8, 9).

$$\mathbf{v}_{10} = \sigma \cdot \frac{\mathbf{b}}{n}, \mathbf{v}_{20} = \sigma \cdot \frac{\mathbf{b}}{n}, \dots, \mathbf{v}_{n0} = \sigma \cdot \frac{\mathbf{b}}{n} \quad (30)$$



Obr. 8: Velikost rezidu v k-tém iteračním kroku s náhodnou proměnnou σ a bez náhodné proměnné σ



Obr. 9: Řešení soustavy $x = (1, 2, 3)^T$ v k-tém iteračním kroku s náhodnou proměnnou σ a bez náhodné proměnné σ

2.3 Maticový zápis algoritmu relaxace úhlu

Při popisu metody relaxace úhlu jsme uvažovali, že se každá relaxace vektoru nevlastního řešení soustavy provádí zvlášť. To můžeme za běžných okolností řešit nějakým podprogramem, který bude tolikrát volaný, kolik existuje proměnných, resp. podle počtu vektorů nevlastního řešení soustavy. Pozn.: tento princip je velmi vhodný pro paralelizaci výpočtu. Abychom udělali algoritmus co možná nejvíce názorný, přepíšeme jej do maticové podoby. Chceme, aby se relaxace úhlu provedla pro všechny vektory najednou, tzn. budeme počítat se všemi vektory, které utvoří dohromady jednu matici. To přispěje k lepší názornosti celého algoritmu. Pro sestavení algoritmu potřebujeme vypočítat důležité proměnné (viz 31-34).

Výpočet normované matice soustavy (31):

$$\mathbf{K} = (\hat{\mathbf{A}}_1 \hat{\mathbf{A}}_2 \dots \hat{\mathbf{A}}_n) \quad (31)$$

Výpočet rezidua. Matice \mathbf{P} nám akumuluje aktuálně dosažené řešení. V prvním kroku iterace je $\mathbf{P} = \mathbf{O}$ (32):

$$\mathbf{r} = \mathbf{b} - \mathbf{P} \cdot \begin{pmatrix} 1 \\ 1 \\ \dots \\ n,1 \end{pmatrix} \quad (32)$$

Výpočet rozdělení rezidua na n vektorů. Tyto vektory sestaví matici \mathbf{V} , která bude zastupovat ve výpočtu jednotlivé vektory nevlastního řešení soustavy jako celek (33):

$$\mathbf{V} = \begin{pmatrix} \frac{\mathbf{r}}{n} & \frac{\mathbf{r}}{n} & \dots & \frac{\mathbf{r}}{n} \end{pmatrix} \quad (33)$$

Výpočet velikosti jednoho vektoru nevlastního řešení soustavy \mathbf{v}_n . Tato velikost je pro všechny vektory z \mathbf{V} stejná (34):

$$l = \left\| \frac{\mathbf{r}}{n} \right\|_e \quad (34)$$

Výpočet znaménkové matice (35). Pro každý relaxovaný vektor musíme určit správný směr (viz vzorec 13, 14, 15). V tomto případě sestavíme znaménkovou matici \mathbf{Z} , která má na diagonále odpovídající hodnoty 1, nebo -1. Znaménková matice vynásobená maticí \mathbf{K} udělí každému sloupci (relaxovanému vektoru z \mathbf{K}) odpovídající znaménko. Matici \mathbf{Z} získáme tak, že vytvoříme dvě matice $(\mathbf{V} + \mathbf{K} \cdot l)^T \cdot (\mathbf{V} + \mathbf{K} \cdot l)$ a $(\mathbf{V} - \mathbf{K} \cdot l)^T \cdot (\mathbf{V} - \mathbf{K} \cdot l)$, které mají na diagonále skalární součin sloupců matice $(\mathbf{V} + \mathbf{K} \cdot l)$ a $(\mathbf{V} - \mathbf{K} \cdot l)$. Porovnání velikostí provedeme odečtením skalárních součinů od sebe a na tento rozdíl (diagonálu) aplikujeme znaménkovou funkci signum. V případě porovnání velikostí bychom měli správně hovořit o porovnání norem. Jelikož se však standardní skalární součin liší oproti Euklidovské normě generované standardním skalárním součinem pouze v odmocnině, můžeme jej pro porovnání stejně tak použít. Je nutné podotknout, že takto provedený výpočet není z hlediska optimalizace času a paměti počítače ideální. Při výpočtu skalárního součinu počítáme také prvky matice

pod a nad diagonálou, které nejsou pro určení znaménkové matice užitečné. Pro urychlení výpočtu počítáme pouze diagonálu matice $(\mathbf{V} + \mathbf{K} \cdot \mathbf{l})^T \cdot (\mathbf{V} + \mathbf{K} \cdot \mathbf{l})$ a $(\mathbf{V} - \mathbf{K} \cdot \mathbf{l})^T \cdot (\mathbf{V} - \mathbf{K} \cdot \mathbf{l})$. Zbylé prvky matice nepočítáme vůbec, nebo je ponecháme nulové.

$$\mathbf{z} = \text{sgn}\left(\text{diag}\left((\mathbf{V} + \mathbf{K} \cdot \mathbf{l})^T \cdot (\mathbf{V} + \mathbf{K} \cdot \mathbf{l}) - (\mathbf{V} - \mathbf{K} \cdot \mathbf{l})^T \cdot (\mathbf{V} - \mathbf{K} \cdot \mathbf{l})\right)\right)$$

$$\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{z}_n \end{pmatrix} \quad (35)$$

Celý iterační krok pak vypadá v maticovém zápisu velice přehledně (36):

$$\mathbf{P}_{k+1} = \sigma \cdot \mathbf{l} \cdot \mathbf{K} \cdot \mathbf{Z} + \mathbf{P}_k \quad (36)$$

Na závěr algoritmu se provede výpočet neznáme \mathbf{x} (37):

$$\mathbf{x} = \begin{pmatrix} \mathbf{P}_{1j} & \mathbf{P}_{2j} & \mathbf{P}_{nj} \\ \mathbf{A}_{1j} & \mathbf{A}_{2j} & \mathbf{A}_{nj} \end{pmatrix}^T \quad (37)$$

Celý algoritmus je pak uveden na obrázku 10.

$$\mathbf{P}_k = \mathbf{O}$$

$$\mathbf{K} = (\widehat{\mathbf{A}}_1 \widehat{\mathbf{A}}_2 \dots \widehat{\mathbf{A}}_n)$$

$$\mathbf{r}_k = \mathbf{b}_k - \mathbf{P}_k \cdot \begin{pmatrix} 1 \\ 1 \\ \dots \end{pmatrix}_{n,1}$$

While (není podmínka zastavení)

$$\mathbf{V}_k = \begin{pmatrix} \frac{\mathbf{r}_k}{n} & \frac{\mathbf{r}_k}{n} & \dots & \frac{\mathbf{r}_k}{n} \end{pmatrix}$$

$$l_k = \left\| \frac{\mathbf{r}_k}{n} \right\|_e$$

$$\mathbf{z}_k = \text{sgn}\left(\text{diag}\left((\mathbf{V}_k + \mathbf{K} \cdot \mathbf{l}_k)^T \cdot (\mathbf{V}_k + \mathbf{K} \cdot \mathbf{l}_k) - (\mathbf{V}_k - \mathbf{K} \cdot \mathbf{l}_k)^T \cdot (\mathbf{V}_k - \mathbf{K} \cdot \mathbf{l}_k)\right)\right)$$

$$\mathbf{Z}_k = \begin{pmatrix} \mathbf{z}_{k1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{z}_{kn} \end{pmatrix}$$

$$\mathbf{P}_{k+1} = \sigma_k \cdot l_k \cdot \mathbf{K} \cdot \mathbf{Z}_k + \mathbf{P}_k$$

$$\mathbf{r}_{k+1} = \mathbf{b}_k - \mathbf{P}_k \cdot \begin{pmatrix} 1 \\ 1 \\ \dots \end{pmatrix}_{n,1}$$

End

$$\mathbf{x} = \begin{pmatrix} \mathbf{P}_{1j} & \mathbf{P}_{2j} & \mathbf{P}_{nj} \\ \mathbf{A}_{1j} & \mathbf{A}_{2j} & \mathbf{A}_{nj} \end{pmatrix}^T$$

Obr. 10 Metoda relaxace úhlu v maticové podobě

2.4 Vlastnosti metody relaxace úhlu

Při popisu vlastností metody relaxace úhlu budeme zkoumat chování algoritmu při řešení vybraných typů soustav lineárních rovnic více proměnných, resp. vybraných matic soustavy. Bude nás zajímat vliv velikosti matice soustavy a čísla podmíněnosti matice soustavy na rychlost konvergence metody. Dále prověříme konvergenci pro obdélníkové, trojúhelníkové, singulární a řídké matice. Pro srovnání budeme řešit stejnou úlohu pomocí již zavedených metod:

- Jacobiova metoda (iterační metoda)
- Gauss-Seidelova metoda (iterační metoda)
- Metoda sdružených gradientů (iterační metoda)
- Řešení pomocí Mooreovy-Penroseovy pseudoinverze (přímá metoda)
- Obecná metoda $x=A\backslash b$ MATLAB (přímá metoda dle typu matice)

Porovnání provedeme v softwaru MATLAB. Jako časová reference nám poslouží obecná metoda implementovaná v softwaru MATLAB pro řešení soustav lineárních rovnic více proměnných (instrukce $x=A\backslash b$). Tato instrukce je maximálně programově optimalizovaná a podle typu matice soustavy vybere vždy nejvhodnější přímý algoritmus řešení. Tím zajistíme časové porovnání s jednou z těch nejlepších obecných metod.

V rámci prvního experimentu jsme nechali vybrané numerické metody řešit soustavy rovnic, u kterých měla matice soustavy vždy konkrétní vlastnosti. Především u iteračních metod má podoba matice soustavy zásadní vliv na konvergenci metody. Jednobodové stacionární iterační metody, které mají iterační předpis $x_{k+1} = B_k \cdot x_k + C_k \cdot b$, potřebují pro konvergenci, aby spektrální poloměr matice iterace B byl menší než jedna, tzn. $\rho(B) < 1$. V našem případě reprezentují tuto třídu Jacobiova a Gauss-Seidelova metoda [3]. Obecně může být výpočetně náročné určit spektrální poloměr matice B , jelikož se musí spočítat vlastní čísla této matice. Je zde ovšem možnost stanovit konvergenci metody také na základě vlastností matice soustavy A . Kupříkladu Jacobiova i Gauss-Seidelova metoda konverguje pro libovolné počáteční přiblížení x_0 , pokud je matice A diagonálně dominantní (38). Gauss-Seidelova metoda konverguje také pro symetrické pozitivně definitní matice (39). Tyto vlastnosti lze ověřit jednodušším způsobem, než je hledání vlastních čísel matice B . Z kategorie projektivních iteračních metod jsme nasadili metodu sdružených gradientů [3], která konverguje pro symetrické pozitivně definitní matice. Toto je velice jednoduchý přehled podmínek konvergence pro námi nasazené iterační metody. Z tohoto důvodu jsme zahrnuli matice s těmito vlastnostmi do našeho experimentu. Najdeme i numerické metody více univerzální, které konvergují i pro nesymetrické lineární systémy, např. metoda bikonjugovaných gradientů [4]. Existují i postupy, jak soustavu převést do požadovaného tvaru, kde matice soustavy je pozitivně definitní. V našem případě jsme porovnávali vybrané metody ve fundamentální podobě, bez jakékoliv úpravy na straně soustavy rovnic.

Prvky diagonálně dominantní matice splňují následující vlastnost (38):

$$|a_{ii}| > \sum_{j=1, j \neq i}^n |a_{ij}| \quad i = 1, 2, \dots, n \quad (38)$$

Matice je pozitivně definitní právě tehdy, když pro každý nenulový vektor \mathbf{x} platí (39):

$$\mathbf{x}^T \cdot \mathbf{A} \cdot \mathbf{x} > 0 \quad (39)$$

Co se týká přímých numerických metod, otestovali jsme Mooreovu-Penroseovu pseudoinverzi (dále jen pseudoinverze) [3], která je postavena na singulárním rozkladu SVD (singular value decomposition) matice soustavy \mathbf{A} . Pseudoinverze je metoda, která je oproti ostatním přímým metodám pomalejší, ale dostatečně robustní, aby si poradila se špatně podmíněnými i obdélníkovými maticemi soustavy \mathbf{A} . Z tohoto důvodu se často používá při výpočtech inverzní kinematické úlohy v robotice.

Pro doplnění zde uvedeme jedno tvrzení z lineární algebry k SVD rozkladu [3]. Nechť \mathbf{A} je reálná matice typu (m, n) , $m \geq n$. Pak existuje matice \mathbf{U} typu (m, m) , diagonální matice \mathbf{W} řádu n s nezápornými diagonálními prvky w_i a čtvercová matice \mathbf{V} řádu n taková, že $\mathbf{A} = \mathbf{U} \cdot \mathbf{W} \cdot \mathbf{V}^T$, přičemž sloupce matice \mathbf{U} jsou navzájem ortonormálními vektory a matice \mathbf{V} je ortogonální. Diagonální prvky matice \mathbf{W} se nazývají singulární čísla matice \mathbf{A} . Jelikož druhé mocniny singulárních čísel matice \mathbf{A} jsou vlastní čísla matice $\mathbf{A}^T \cdot \mathbf{A}$, převádí se problém hledání SVD rozkladu na hledání vlastních čísel a vlastních vektorů matice $\mathbf{A}^T \cdot \mathbf{A}$. Jako příklad numerického hledání SVD rozkladu můžeme uvést algoritmus G. H. Goluba and C. Reinsche [3].

Pro regulární matici \mathbf{A} , kde všechna vlastní (singulární) čísla jsou nenulová, platí jednoduchý vztah pro inverzní matici, který vychází z SVD rozkladu $\mathbf{A}^{-1} = \mathbf{V} \cdot \mathbf{W}^{-1} \cdot \mathbf{U}^T$. V případě singulární matice, kde alespoň jedno vlastní (singulární) číslo je nenulové, nelze provést inverzi diagonální matice \mathbf{W} . V případě, že $w_i = 0$, nutně bychom dělili nulou. V tomto případě se používá právě pseudoinverze $\mathbf{A}^+ = \mathbf{V} \cdot \mathbf{W}^l \cdot \mathbf{U}^T$, kde při stanovení diagonály $\mathbf{W}^l = \text{diag}(p_i)$ platí 40, 41.

$$p_i = \frac{1}{w_i}, \text{ pokud } w_i \neq 0 \quad (40)$$

$$p_i = 0, \text{ pokud } w_i = 0 \quad (41)$$

Pseudoinverzi lze sestavit jak pro singulární, tak pro regulární matici \mathbf{A} . Pokud je matice \mathbf{A} regulární, pak $\mathbf{W}^{-1} = \mathbf{W}^l$. V případě, že je matice \mathbf{A} singulární a řešení soustavy lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ má nekonečně mnoho řešení, pak $\mathbf{x} = \mathbf{A}^+ \cdot \mathbf{b}$, kde $\|\mathbf{x}\|_e \leq \|\mathbf{y}\|_e$. Pseudoinverze nalezne řešení nejmenší v euklidovské normě ze všech možných řešení. V případě, že je matice \mathbf{A} singulární a řešení soustavy lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ nemá řešení, pak $\mathbf{x} = \mathbf{A}^+ \cdot \mathbf{b}$, kde $\|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}\|_e \leq \|\mathbf{b} - \mathbf{A} \cdot \mathbf{y}\|_e$. Pseudoinverze nalezne řešení s nejmenším reziduem v euklidovské normě. Jak již bylo jednou řečeno,

pseudoinverze je dostatečně robustní, aby si poradila se špatně podmíněnými i obdélníkovými maticemi soustavy \mathbf{A} , což je vidět na výsledcích našeho prvního experimentu (viz tab. 1).

Tab. 1: Porovnání konvergence vybraných numerických metod

Vlastnosti	Matice soustavy										Rozměr
	Involutorní	Cauchyho	Tridiagonální (s nuly na diagonále)	Hankelova	Horní trapezoidální	Přeurčená	Nedourčená	Lehmerova	Dorrova	Poissonova	
Pozitivně definitní	Ne	Ano	Ne	Ano	Ano	Ne	Ne	Ano	Ano	Ano	m=n
Diagonálně dominantní	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ano	Ano	
Řidká	Ne	Ne	Ano	Ne	Ne	Ne	Ne	Ne	Ne	Ne	
Nuly na diagonále	Ne	Ne	Ano	Ne	Ne	Ne	Ne	Ne	Ne	Ne	
Symetrická	Ne	Ano	Ano	Ano	Ne	Ne	Ne	Ano	Ne	Ano	
Trojúhelníková	Ne	Ne	Ne	Ne	Ano	Ne	Ne	Ne	Ne	Ne	
Tridiagonální	Ne	Ne	Ano	Ne	Ne	Ne	Ne	Ne	Ano	Ne	
Regulární	Ano	Ano	Ano	Ano	Ano	Ne	Ne	Ano	Ano	Ano	
Obdélníková	Ne	Ne	Ne	Ne	Ne	Ano	Ne	Ne	Ne	Ne	m>n
	Ne	Ne	Ne	Ne	Ne	Ne	Ano	Ne	Ne	Ne	n<m
Metoda	Konvergence pro jednotlivé numerické metody										Typ
Jacobiova metoda	Ne	Ne	Ne	Ne	Ano	Ne	Ne	Ne	Ano	Ano	Iterační
Gauss-Seidelova metoda	Ne	Ano	Ne	Ano	Ano	Ne	Ne	Ano	Ano	Ano	Iterační
Metoda sdružených gradientů	Ne	Ano	Ano	Ano	Ano	Ne	Ne	Ano	Ne	Ano	Iterační
Pseudoinverze	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Přímá
Metoda relaxace úhlu	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Iterační

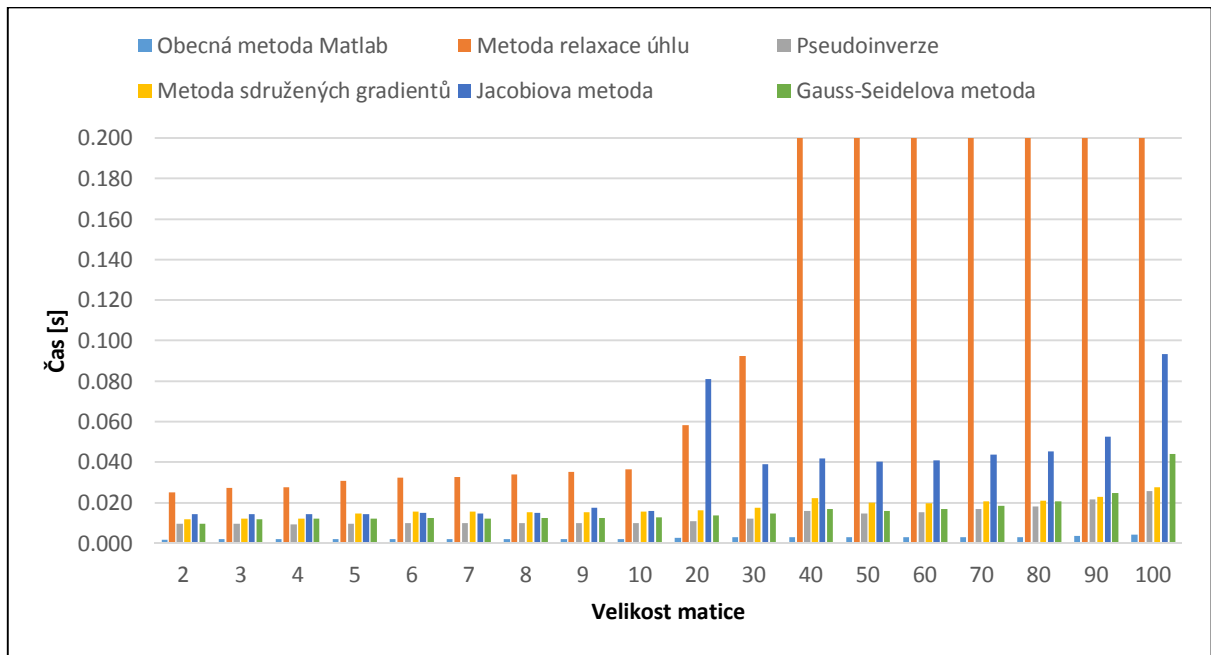
V tabulce 1 můžeme vidět porovnání vybraných numerických metod a jejich úspěšnost (konvergenci) pro matice soustavy s předem definovanými vlastnostmi. Pro každou matici soustavy je vždy uveden seznam vlastností a informace o konvergenci vybrané metody. Můžeme si všimnout, že metody konvergují dle výše uvedených předpokladů. V tomto testu obstála velmi dobře nová metoda relaxace úhlu, která konverguje ve všech případech bez ohledu na vlastnosti matice soustavy. Tzn. i pro obdélníkové matice. To není u iteračních metod běžné. Můžeme konstatovat, že co do robustnosti řešení dopadla metoda relaxace úhlu stejně dobře jako pseudoinverze. V tomto testu jsme nesledovali rychlost konvergence, ale pouze schopnost přiblížit se k řešení bez ohledu na čas.

Časové porovnání vybraných metod jsme provedli v následujícím experimentu. Aby konvergovaly všechny metody, pracovali jsme výhradně s diagonálně dominantními maticemi. Postupně jsme zvětšovali velikost matice a měřili čas potřebný pro vyřešení soustavy lineárních rovnic. Vždy existovalo pouze jedno řešení soustavy $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) = n$. Iterační metody jsme zastavili, pokud norma rezidua $\|\mathbf{r}\|_e$ byla menší jak 0.01, nebo počet iterací dosáhl 20 000. Pro každou velikost matice jsme provedli výpočet pro sto náhodně vygenerovaných soustav lineárních rovnic. V tabulce 2

jsou uvedeny průměrné hodnoty ze všech měření pro danou velikost matice soustavy. Na obrázku 11 jsou znázorněny průměrné časy v grafu.

Tab. 2: Časové porovnání vybraných numerických metod pro čtvercové matice

Matice m=n			Ø Čas [s]						Ø Počet iterací			
m	n	Číslo podmíňnosti	Obecná metoda $x=A\backslash b$ Matlab	Metoda relaxace úhlu	Pseudoinverze	Metoda sdružených gradientů	Jacobiova metoda	Gauss-Seidelova metoda	Metoda relaxace úhlu	Metoda sdružených gradientů	Jacobiova metoda	Gauss-Seidelova metoda
2	2	2.455	0.002	0.025	0.010	0.012	0.014	0.010	36	3	8	5
3	3	3.236	0.002	0.027	0.010	0.012	0.015	0.012	61	4	10	6
4	4	4.203	0.002	0.028	0.010	0.012	0.015	0.012	88	5	12	7
5	5	4.726	0.002	0.031	0.010	0.015	0.015	0.012	115	6	16	8
6	6	5.485	0.002	0.032	0.010	0.016	0.015	0.013	141	7	19	9
7	7	5.299	0.002	0.033	0.010	0.016	0.015	0.012	171	8	21	9
8	8	6.376	0.002	0.034	0.010	0.015	0.015	0.013	199	9	27	10
9	9	6.199	0.002	0.035	0.010	0.015	0.018	0.013	224	10	231	11
10	10	7.013	0.002	0.037	0.010	0.016	0.016	0.013	254	11	82	11
20	20	9.813	0.003	0.059	0.011	0.016	0.081	0.014	553	21	4273	18
30	30	12.457	0.003	0.093	0.012	0.018	0.039	0.015	892	31	1109	26
40	40	15.335	0.003	0.223	0.016	0.022	0.042	0.017	1456	38	748	34
50	50	18.615	0.003	0.433	0.015	0.020	0.041	0.016	2173	46	587	45
60	60	21.318	0.003	0.706	0.016	0.020	0.041	0.017	2941	51	518	53
70	70	24.527	0.003	1.154	0.017	0.021	0.044	0.019	3947	57	458	67
80	80	28.056	0.003	1.777	0.018	0.021	0.045	0.021	5057	65	422	78
90	90	31.106	0.004	3.456	0.022	0.023	0.053	0.025	6238	72	393	94
100	100	35.357	0.004	6.193	0.026	0.028	0.094	0.044	8007	82	370	113

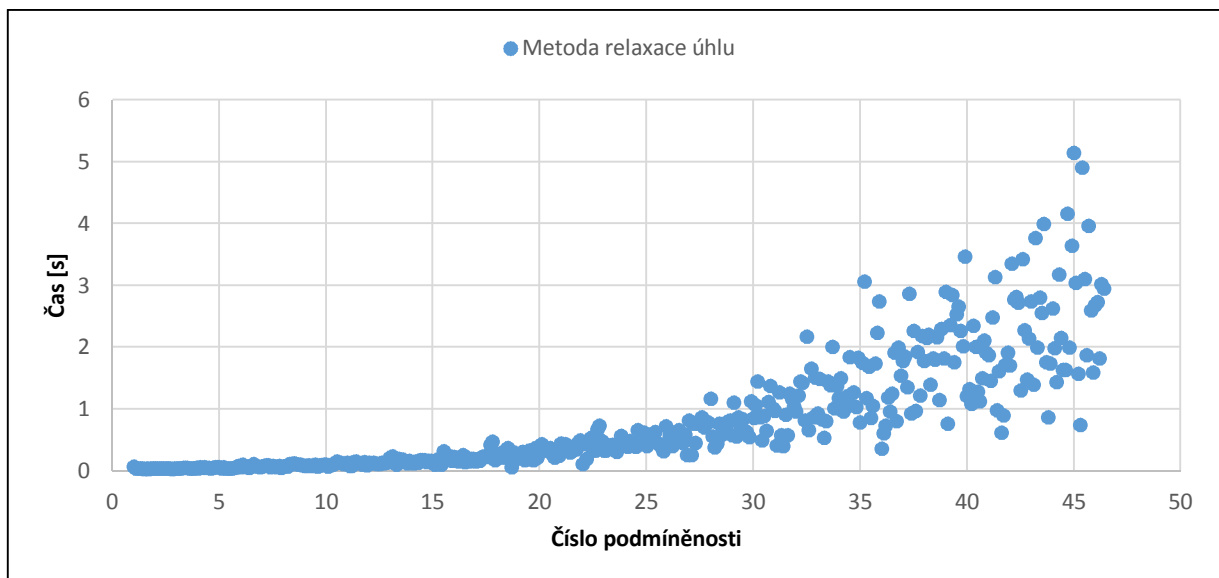


Obr. 11 Čas potřebný pro vyřešení soustavy lineárních rovnic vybraných numerických metod

Dále jsme testovali vliv čísla podmíněnosti matice soustavy (42) na rychlost konvergence metody relaxace úhlu (viz. obr. 12). Pro stejně velkou soustavu lineárních rovnic $n = 10$ jsme generovali matice soustavy se stále větším číslem podmíněnosti. Zastavení metody bylo opět podmíněno normou rezidua $\|r\|_e < 0.01$, nebo dosažením počtu iterací 20 000.

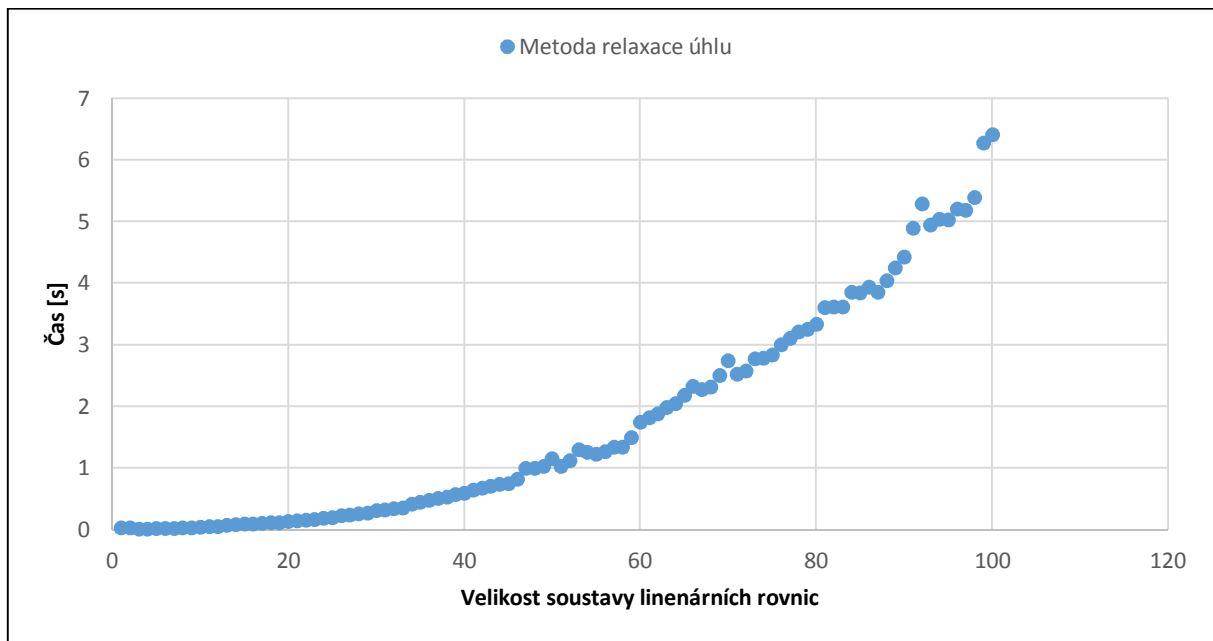
Číslo podmíněnosti se určuje v softwaru MATLAB dle vztahu 42. Pseudoinverze umožňuje stanovit číslo podmíněnosti i pro singulární matice, které nejsou invertovatelné.

$$\kappa(A) = \|A\|_e \cdot \|A^+\|_e \quad (42)$$



Obr. 12 Vliv čísla podmíněnosti matice soustavy na rychlost konvergence metody relaxace úhlu, $n = 10$

Zásadní vliv na rychlost konvergence metody relaxace úhlu má také velikost soustavy lineárních rovnic. Tento vliv jsme testovali postupným generováním stále větší soustavy lineárních rovnic, kde matice soustavy měla stále stejné číslo podmíněnosti. To bylo rovno jedné, tzn. nejlepší možná hodnota. Tímto způsobem jsme zabránili vlivu čísla podmíněnosti na rychlost konvergence v tomto experimentu a měřili jsme hlavně čas ovlivněný velikostí soustavy. Rychlost konvergence metody relaxace úhlu v závislosti na velikosti soustavy lineárních rovnic je vidět na obrázku 13.



Obr. 13: Vliv velikosti soustavy lineárních rovnic na rychlosti konvergence metody relaxace úhlu, $\kappa(A) = 1$

Do této chvíle jsme pracovali pouze s čtvercovými regulárními maticemi, kde soustava lineárních rovnic měla vždy jedno řešení. Jelikož metoda relaxace úhlu umí nalézt řešení pro obdélníkové matice, provedli jsme časové porovnání i pro tento typ soustav lineárních rovnic (viz tab. 3). Pro porovnání jsme ponechali pouze metodu relaxace úhlu a vybrané přímé metody, jelikož zbylé iterační metody pro obdélníkové matice nelze použít. Opět jsme generovali různě velké matice a pro sto náhodných případů soustav lineárních rovnic jsme stanovili průměrný čas řešení. V případě, že matice soustavy byla přeuročená $m > n$, existovalo pouze jedno řešení soustavy lineárních rovnic. V případě, že matice soustavy byla nedostatečně určená $m < n$, existovalo nekonečně mnoho řešení soustavy lineárních rovnic. Podmínky zastavení iterační metody relaxace úhlu jsme opět ponechali stejné, tzn. $\|r\|_e < 0.01$, nebo 20 000 iterací.

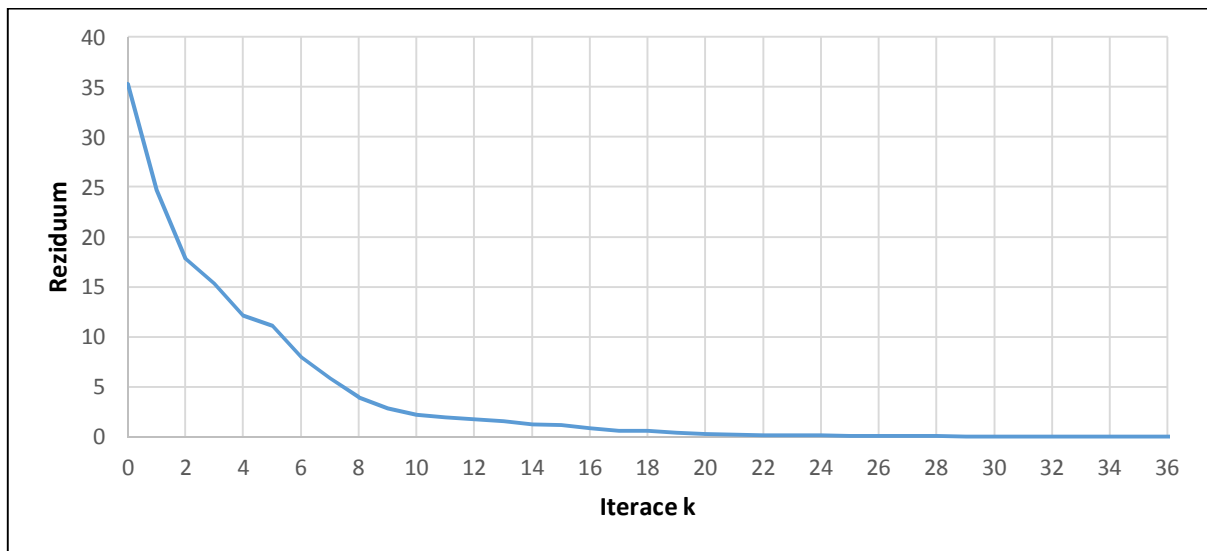
Tab. 3: Časové porovnání vybraných numerických metod pro obdélníkové matice

Matice $m < n$			Ø Čas [s]			Matice $m > n$			Ø Čas [s]		
m	n	Číslo podmíňnosti	Obecná metoda $x=A\backslash b$ Matlab	Metoda relaxace úhlu	Pseudoinvertze	m	n	Číslo podmíňnosti	Obecná metoda $x=A\backslash b$ Matlab	Metoda relaxace úhlu	Pseudoinvertze
2	3	2.557	0.002	0.029	0.011	3	2	2.371	0.002	0.026	0.010
3	4	3.440	0.002	0.027	0.010	4	3	3.476	0.002	0.027	0.010
4	5	4.227	0.002	0.030	0.010	5	4	4.100	0.002	0.028	0.010
5	6	5.034	0.002	0.031	0.010	6	5	4.666	0.002	0.031	0.010
6	7	5.568	0.002	0.032	0.010	7	6	5.322	0.002	0.031	0.010
7	8	5.789	0.002	0.033	0.010	8	7	5.950	0.002	0.032	0.010
8	9	6.132	0.002	0.034	0.010	9	8	6.667	0.002	0.034	0.010
9	10	6.868	0.002	0.035	0.010	10	9	6.736	0.002	0.035	0.010
10	11	7.225	0.002	0.036	0.010	11	10	6.912	0.002	0.037	0.010
20	21	10.095	0.002	0.052	0.010	21	20	9.579	0.002	0.050	0.010
30	31	12.322	0.002	0.081	0.011	31	30	12.661	0.003	0.083	0.011
40	41	15.737	0.003	0.222	0.015	41	40	14.872	0.003	0.211	0.016
50	51	18.345	0.004	0.470	0.015	51	50	18.564	0.003	0.449	0.015
60	61	21.848	0.004	0.833	0.016	61	60	21.372	0.003	0.761	0.016
70	71	25.128	0.004	1.372	0.018	71	70	24.914	0.004	1.294	0.018
80	81	28.283	0.004	2.384	0.021	81	80	28.310	0.004	2.058	0.020
90	91	31.901	0.004	3.779	0.022	91	90	31.678	0.004	3.458	0.021
100	101	35.316	0.004	6.356	0.024	101	100	34.946	0.004	6.230	0.025

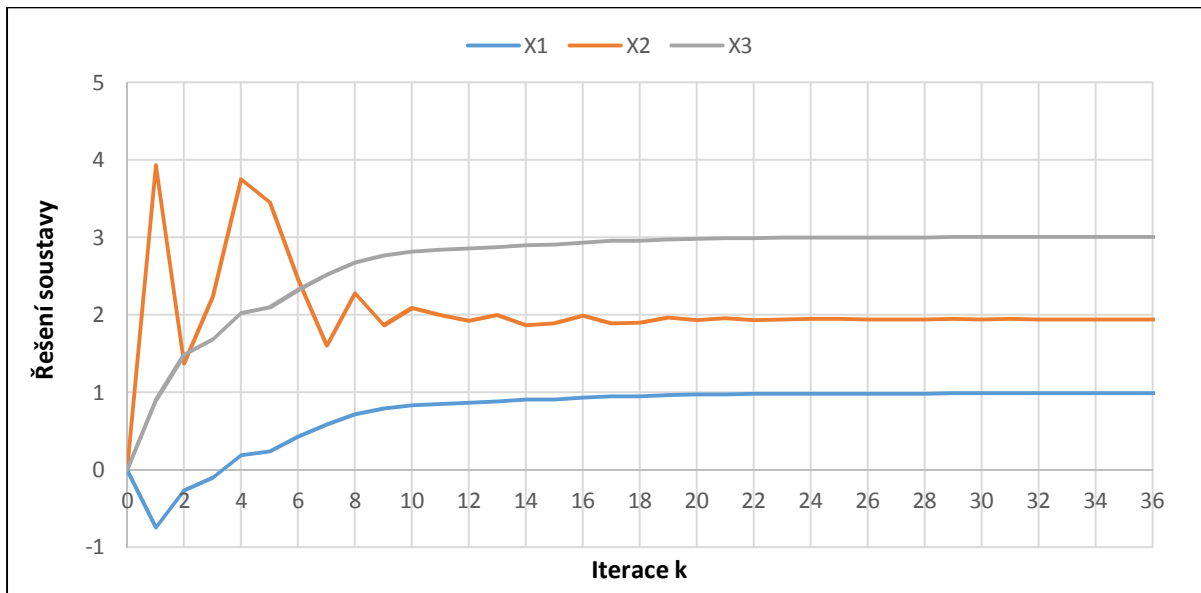
V tuto chvíli můžeme shrnout dosažené výsledky z našeho druhého experimentu. V rámci porovnání času potřebného k vyřešení soustavy lineárních rovnic dopadla metoda relaxace úhlu ze všech vybraných numerických metod nejhůře. Nárůst času v závislosti na velikosti matice a číslu podmíňnosti je u metody relaxace úhlu tak razantní, že se velice rychle dostáváme do řádu sekund. Přijatelně lze použít metodu relaxace úhlu pro matice zhruba do velikosti $m, n \leq 10$, kde se pohybuje časová náročnost v rámci několika desítek milisekund. To je stále dvakrát až třikrát více, než u ostatních iteračních metod. Oproti dobrým (obecným) přímým metodám je rozdíl ještě větší (viz obr. 11). Z toho důvodu nemůžeme doporučit metodu relaxace úhlu jako jednu z obecných iteračních metod pro řešení soustav lineárních rovnic. V obecném případě bychom použili pro malé soustavy lineárních rovnic nějakou přímou metodu. Přímá metoda bude totiž pro malé soustavy lineárních rovnic daleko rychlejší, než metoda relaxace úhlu. Pro velké soustavy lineárních rovnic zase není možné metodu relaxace úhlu kvůli nárůstu výpočetního času použít. V tomto případě použijeme raději vybranou iterační metodu. Z hlediska dosažených časových výsledků není vidět velký rozdíl mezi případem se singulární a regulární maticí soustavy. V obou případech je zde stejný problém s nárůstem výpočetního času. Zásadní vliv na nárůst času u metody relaxace úhlu má fakt, že v každém iteračním kroku musíme

provést n relaxací úhlu. Tento počet bude tím větší, čím bude větší velikost soustavy. Pro snížení časové náročnosti metody se zde nabízí možnost paralelizace výpočtu, jelikož můžeme každou relaxaci úhlu počítat nezávisle. Proti zhoršené konvergenci u špatně podmíněných soustav můžeme nasadit známé techniky předpodmiňování, např. Jacobiho předpodmiňování [3], nekompletní Cholského faktorizaci [3] apod..

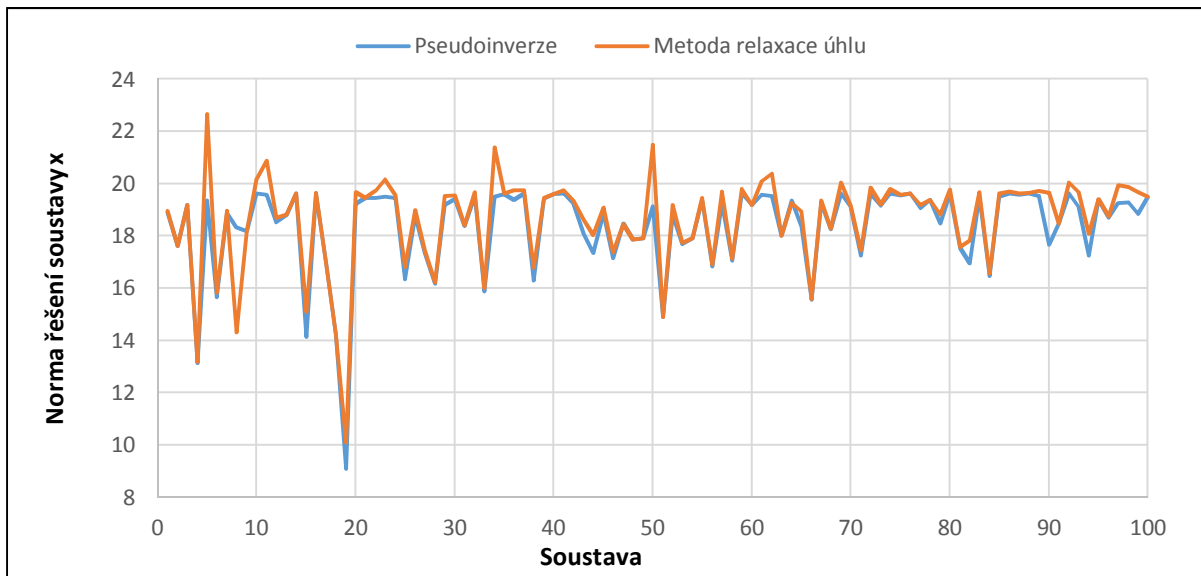
Nabízí se zde otázka, k jakému řešení konverguje metoda relaxace úhlu v případě, kdy má soustava lineárních rovnic nekonečně mnoho řešení $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) < n$, a v případě, kdy žádné řešení nemá $h(\mathbf{A}) < h(\mathbf{A}|\mathbf{b})$. V rámci testu s obdélníkovými maticemi jsme vyzkoušeli, že se metoda relaxace úhlu snaží v obou případech minimalizovat reziduum. V případě, kdy má soustava nekonečně mnoho řešení, se zbytkové reziduum blíží k nule (viz obr. 14). Průběh řešení se postupně ustálí (viz obr. 15). Ukazuje se také, že norma tohoto řešení se blíží k normě, kterou nalezneme pomocí pseudoinverze, tzn. řešení nejmenší v euklidovské normě $\|\mathbf{x}_{RLX}\|_e \approx \|\mathbf{x}_{PSD}\|_e \leq \|\mathbf{y}\|_e$. Porovnání velikosti této normy pro obě metody pro sto náhodně vygenerovaných soustav lineárních rovnic o velikosti $n = 10$ s nekonečně mnoho řešeními je znázorněno na obrázku 16. Na tomto obrázku můžeme vidět, že metoda relaxace úhlu kopíruje co do velikosti $\|\mathbf{x}_{RLX}\|_e$ hodnoty, které bychom dostali pomocí pseudoinverze $\|\mathbf{x}_{PSD}\|_e$. Jsou zde samozřejmě odchylky, jelikož metoda relaxace úhlu se k hodnotě nulového rezidua pouze blíží. Proto nedostaneme po konečném počtu iterací absolutně přesné řešení, a tedy stejně velkou normu. Také výsledné řešení se u obou metod může hodnotově výrazně lišit i pro téměř stejnou normu, tzn. $\mathbf{x}_{RLX} \neq \mathbf{x}_{PSD}$ pro $\|\mathbf{x}_{RLX}\|_e \approx \|\mathbf{x}_{PSD}\|_e$. Je to dáno tím, že pracujeme se singularní maticí soustavy, kde každá malá nepřesnost, nebo zaokrouhlení, může způsobit velký rozdíl v řešení soustavy.



Obr. 14: Velikost rezidua v k -tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic s nekonečně mnoho řešeními



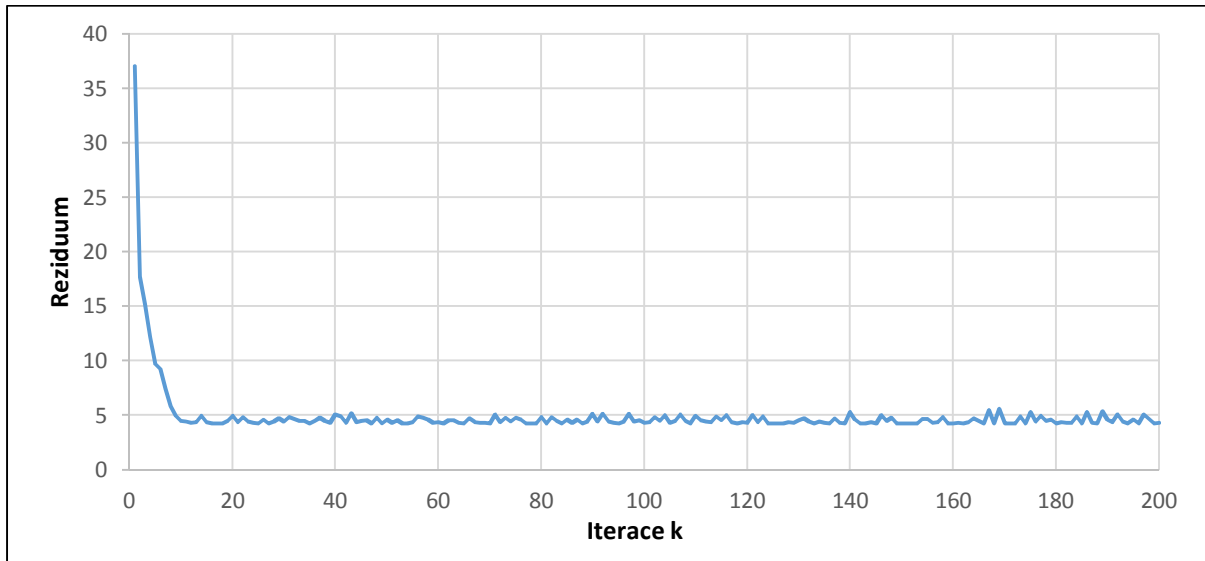
Obr. 15: Řešení soustavy $(x_1, x_2, x_3)^T$ v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic s nekonečně mnoho řešeními



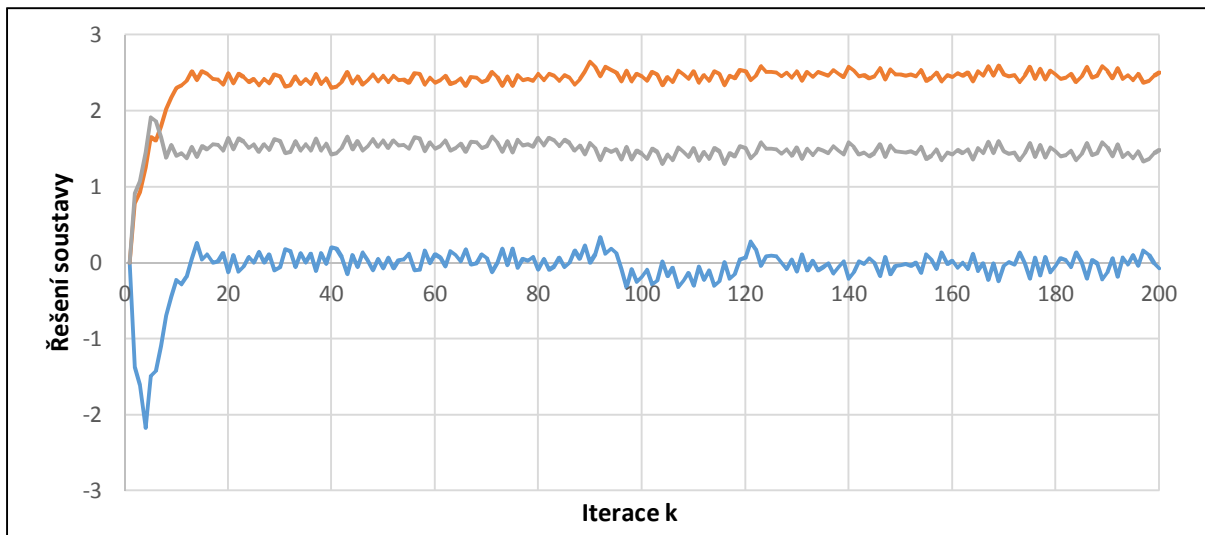
Obr. 16: Norma $\|x\|_e$ pro sto náhodně generovaných soustav lineárních rovnic s nekonečně mnoho řešeními

V případě, kdy soustava lineárních rovnic nemá žádné řešení, se metoda relaxace úhlu snaží minimalizovat reziduum. Není ovšem schopná dosáhnout nulové hodnoty rezidua a osciluje kolem minimální hranice (viz obr. 17). Dosáhnout nulové hodnoty ani nemůže, jelikož žádné přesné řešení soustavy neexistuje. Průběh řešení se postupně ustálí a osciluje kolem určité hodnoty (viz obr. 18). I v tomto případě jsme provedli test porovnání norem pro sto náhodně vygenerovaných soustav $n = 10$. Nejde však o porovnání euklidovské normy řešení soustavy, ale o euklidovské normy rezidua $\|b - A \cdot x\|_e$ (obr. 19). Na obrázku 19 je jasně patrné, že metoda relaxace úhlu kopíruje hodnoty rezidua, které získáme pomocí pseudoinverze. Na základě tohoto experimentu můžeme usuzovat, že i v případě soustavy lineárních rovnic, které nemají jasné řešení, se bude metoda relaxace úhlu s největší pravděpodobností blížit k řešení s nejmenším reziduem stejně jako pseudoinverze $\|b - A \cdot x_{RLX}\|_e \approx$

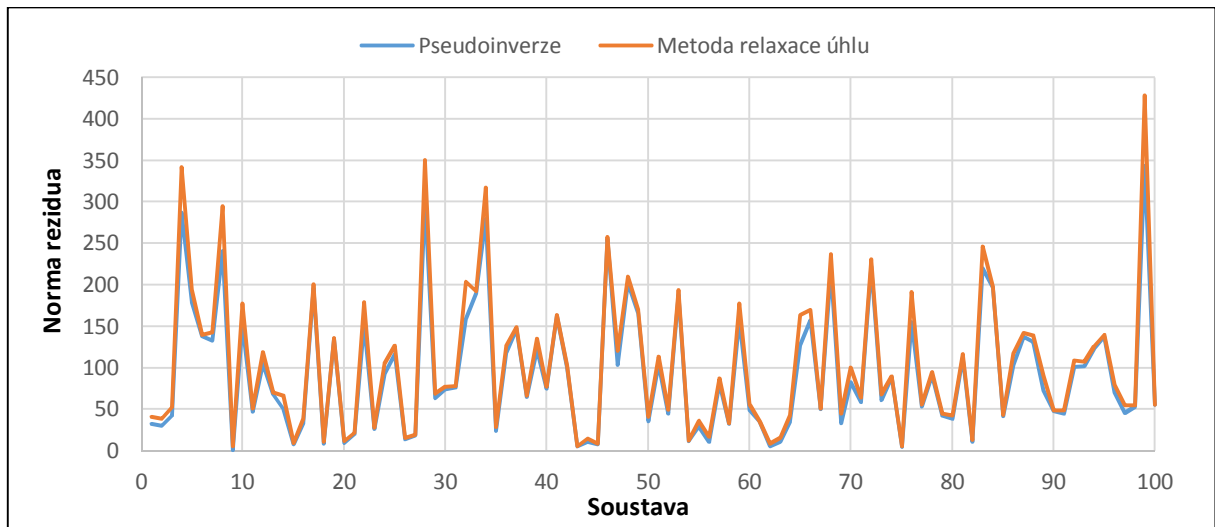
$\|b - A \cdot x_{PSD}\|_e \leq \|b - A \cdot y\|_e$. Řešení soustavy získané metodou relaxace úhlu opět nemusí být zcela identické s řešením, které získáme pomocí pseudoinverze i pro stejnou velkou normu rezidua, tzn. $x_{RLX} \neq x_{PSD}$ pro $\|b - A \cdot x_{RLX}\|_e \approx \|b - A \cdot x_{PSD}\|_e$. Velký vliv má právě oscilace algoritmu. Proto se řešení soustavy mění i při relativním ustálení rezidua kolem minimální hodnoty. V příkladu z obrázku 17 vyšlo řešení $x_{PSD} = (-0.0649, 2.5065, 1.3766)$ a $x_{RLX} = (-0.0703, 2.5013, 1.4891)$, což jsou hodnoty relativně blízké. Uvedené x_{RLX} je z posledního kroku iterace. V jiném iteračním kroku se může hodnota x_{RLX} lišit dle průběhu až o desetiny.



Obr. 17: Velikost rezidua v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic bez řešení



Obr. 18: Řešení soustavy v k-tém iteračním kroku metody relaxace úhlu pro soustavu lineárních rovnic bez řešení



Obr. 19: Norma $\|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}\|_e$ pro sto náhodně generovaných soustav lineárních rovnic bez řešení

V tuto chvíli máme odzkoušeny základní vlastnosti metody relaxace úhlu. Je nutné konstatovat, že všechny vlastnosti byly ověřeny experimentálně na dostatečném množství případů soustav lineárních rovnic. Teoretickými důkazy dílčích vlastností, kromě samotné konvergence, jsme se v rámci této disertační práce nezabývali. Další zkoumání metody by bylo vhodné provést odborníkem z oblasti matematiky, který může výše popsané vlastnosti dále ověřit a objasnit příčiny tohoto chování.

Na základě našeho zkoumání má metoda relaxace úhlu následující vlastnosti:

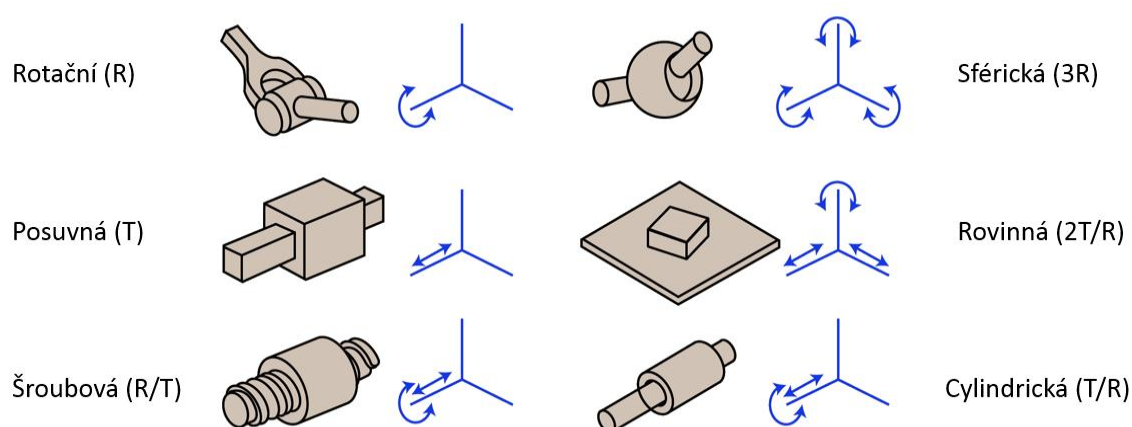
- pokud má soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ jedno řešení $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) = n$, tak metoda relaxace úhlu konverguje k tomuto řešení,
- pokud má soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ nekonečně mnoho řešení $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) < n$, tak se metoda relaxace úhlu blíží k řešení nejmenší v euklidovské normě $\|\mathbf{x}_{RLX}\|_e \approx \|\mathbf{x}\|_e \leq \|\mathbf{y}\|_e$,
- pokud soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ nemá řešení $h(\mathbf{A}) < h(\mathbf{A}|\mathbf{b})$, tak se metoda relaxace úhlu blíží k řešení s nejmenším reziduem v euklidovské normě $\|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}_{RLX}\|_e \approx \|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}\|_e \leq \|\mathbf{b} - \mathbf{A} \cdot \mathbf{y}\|_e$,
- výpočet funguje i pro obdélníkové matice,
- metoda relaxace úhlu vykazuje tak vysoký nárůst výpočetního času v závislosti na velikosti matice a číslu podmíněnosti matice soustavy, že se nedá uvažovat jako nová obecná metoda pro výpočet soustav lineárních rovnic.

3 Výpočet inverzní kinematické úlohy v robotice

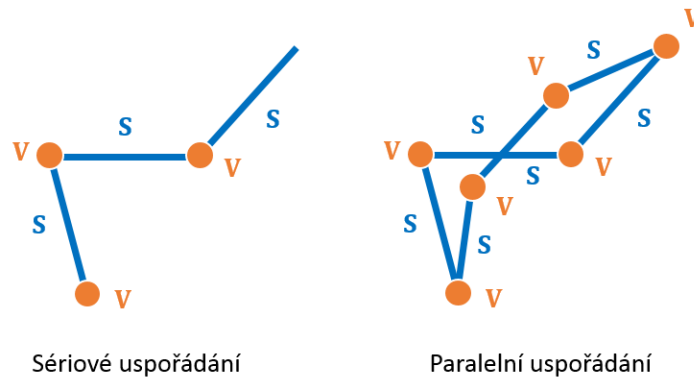
Cílem další části disertační práce bude oblast robotiky a výpočet inverzní kinematické úlohy v robotice na základě metody relaxace úhlu. Nenecháme se odradit špatnými časovými výsledky metody v rámci řešení soustav lineárních rovnic a pokusíme se využít jejich lepších vlastností. Budeme vycházet z faktu, že metoda relaxace úhlu má co do výsledků obdobné řešení jako pseudoinverze a stejně jako pseudoinverze je schopná pracovat s obdélníkovými maticemi. Můžeme si představit, že se jedná o určitou iterační obdobu pseudoinverze, která výsledné řešení odhadne. Ukážeme si, že v rámci výpočtu inverzní kinematické úlohy v robotice matice nedosahují takových velikostí a více než rychlost je důležitá robustnost algoritmu vůči měnící se podobě matice soustavy pro různé postavení robota. Právě z tohoto důvodu se v současné praxi hojně využívá pseudoinverze, ačkoliv může být vůči jiným přímým metodám pomalejší. Pokusíme se tedy používanou pseudoinverzi nahradit metodou relaxace úhlu a ukážeme si vlastnosti tohoto řešení.

3.1 Kinematická struktura a její popis

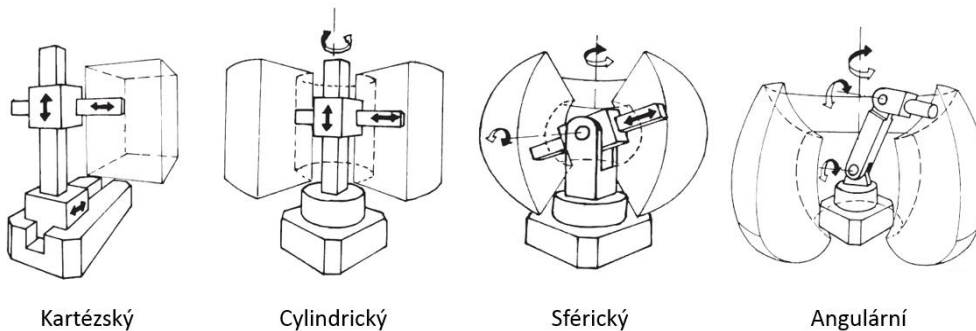
Kinematická struktura robota se skládá z jednotlivých mechanických částí, které můžeme nazvat spoje, nebo také ramena (v této práci budeme užívat výraz spoje). V případě simulace, nebo animace, to mohou být virtuální spoje. Jednotlivé spoje kinematické struktury mohou být propojeny různými kinematickými vazbami, které dovolují kombinaci translačně rotačního pohybu (viz obr. 20). Z jednotlivých spojů a vazeb lze poskládat celou kinematickou strukturu, která může být v sériové nebo paralelní podobě (viz obr. 21). Každá kinematická struktura má dle použitých vazeb určitý rozsah pohybu, který definuje její pracovní prostor. Příklady několika základních kinematických struktur i s pracovními prostory jsou na obrázku 22.



Obr. 20: Přehled kinematických vazeb (R – Rotace, T – Translace), obrázek převzat z [30]



Obr. 21: Příklad sériového a paralelního uspořádání kinematické struktury (V – Vazba, S – Spoj)

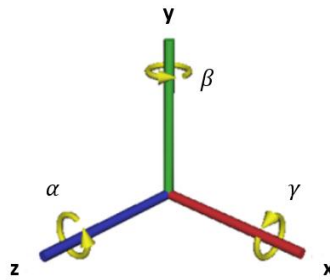


Obr. 22: Pracovní prostor kinematické struktury dle typu robota, obrázek převzat z [31]

Aby se kinematická struktura mohla pohybovat, je nutné zapracovat do jednotlivých vazeb nějaký pohon. Toto se netýká třeba oblasti animace, kde nás realizace pohybu nelimituje. V praxi se často používá pro reálné stroje rotační a posuvná vazba, jelikož se dá velmi dobře spojit přes hřídel s motorem. Kvůli případné regulaci pohybu je zde možnost odměřování polohy hřídele standardními metodami, např. pomocí absolutního enkodéru, resolveru apod.. Stejně tak tomu je i v případě průmyslových robotů, které jsou postaveny především na rotačních vazbách.

V rámci této disertační práce budeme uvádět příklady právě pro kinematické struktury s rotačními vazbami. Každá rotační vazba bude definovaná svou polohou (úhlem natočení) v prostoru kloubových souřadnic $\theta = (\theta_1, \theta_2, \dots, \theta_n)$. Poloha koncového bodu (efektoru) kinematické struktury bude popsána v prostoru koncového bodu T . Prostor koncového bodu si lépe definujeme. V našem případě se jedná o popis koncového bodu pomocí translace a rotace v kartézském souřadnicovém systému robota B_R pomocí šesti parametrů $T_R = [x, y, z, \alpha, \beta, \gamma]$. Parametry $[x, y, z]$ popisují translaci a parametry $[\alpha, \beta, \gamma]$ rotaci souřadného systému vůči B_R , kde α je rotace kolem osy z , β je rotace kolem osy y a γ je rotace kolem osy x (viz obr. 23) [32]. Jelikož se jedná o popis pozice bodu v prostoru, budeme rotaci společně s translací uzavírat do hraných závorek. Kromě translace a rotace lze také vycházet např. z popisu pomocí kvaternionů [17]. Tento popis zde uvádět nebudeme. Na základě translace a rotace si člověk dokáže dobře představit pozici nějakého bodu v prostoru a jeho natočení. Proto se tento zápis používá při simulaci robotových trajektorií v CAD systémech. Tento zápis není

ovšem vhodný při výpočtech dílčích transformací, pro které se lépe hodí zápis pomocí homogenní matice transformace T_H (43). Oba dva zápisy lze vzájemně přepočítat $T_H \leftrightarrow T_R$. Výpočet homogenní matice transformace na základě známé translace a rotace je popsán pomocí rovnice 43-45. Výpočet matice rotace R je dán sekvencí $R_z \cdot R_y \cdot R_x$, což je běžný průmyslový standard. První tři sloupcečky homogenní matice transformace pak odpovídají směru (vektoru) jednotlivých os souřadného systému v pořadí x, y, z (viz barevné značení rovnice 43 a obr. 23). Lze použít i jinou libovolnou sekvenci výpočtu matice rotace R , např. $R_x \cdot R_y \cdot R_z$, ale jelikož násobení matic není komutativní, interpretace homogenní matice transformace by pro tento případ nebyla stejná. V této práci zůstaneme u sekvence, která je v průmyslu běžná. Zpětný převod z T_H na T_R zde uvádět nebudeme. Pro námi použitou matici rotace je metoda zpětného převodu popsána např. zde [11]. Homogenní matice transformace nám umožní vyjádřit výslednou homogenní matici transformace T_H jako součin dílčích transformací $T_1 \dots T_n$ mezi jednotlivými vazbami kinematického řetězce. Potom $T_H = B_H \cdot T_1 \cdot T_2 \cdot T_3 \cdot \dots \cdot T_n$. Jelikož $B_R = [0,0,0,0,0,0]$ je pro nás výchozí souřadný systém robota, potom $B_H = I$ a výsledné $T_H = T_1 \cdot T_2 \cdot T_3 \cdot \dots \cdot T_n$.



Obr. 23: Rotace kolem souřadného systému popsána parametry $[\alpha \beta \gamma]$, obrázek převzat z [32]

$$T_H = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (43)$$

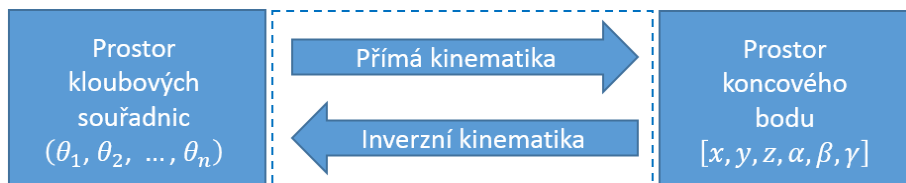
$$P = [x \ y \ z]^T \quad (44)$$

$$R = R_z \cdot R_y \cdot R_x$$

$$R = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix} \quad (45)$$

3.2 Přímá kinematická úloha v robotice

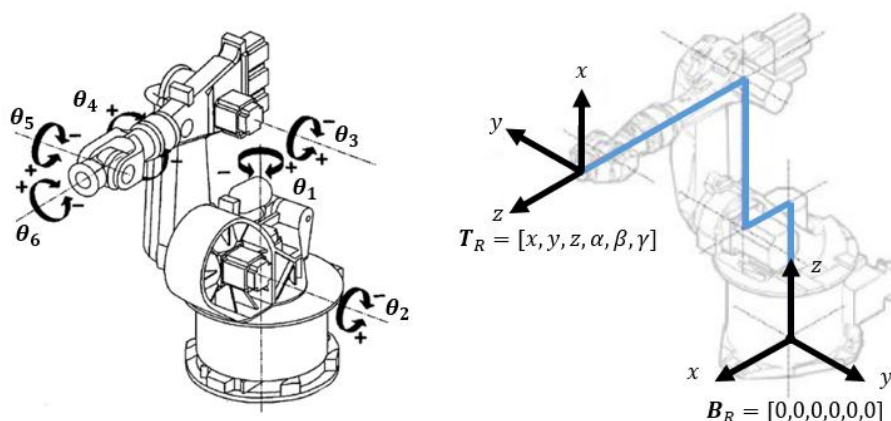
Přímá a inverzní kinematická úloha v robotice patří mezi základní problematiku, která před námi stojí při popisu kinematických vazeb [1], při řešení mechanických manipulátorů [2], nebo při animaci pohybu ve filmovém a herním průmyslu [3-4]. Než přistoupíme k řešení inverzní kinematické úlohy, popíšeme si nejprve úlohu přímou. Inverzní kinematická úloha totiž vychází z rovnic, které získáme právě popisem úlohy přímé. Vzájemná souvislost přímé a inverzní kinematické úlohy je znázorněna na obrázku 24.



Obr. 24: Přímá a inverzní kinematická úloha v robotice

Přímá kinematická úloha v robotice [9] je zobrazení z prostoru kloubových souřadnic θ do prostoru polohy koncového bodu kinematické struktury T . To znamená, že známe polohu všech, nebo některých, rotačních vazeb $\theta = (\theta_1, \theta_2, \dots, \theta_n)$ a hledáme polohu koncového bodu v kartézském souřadnicovém systému T_R , resp. homogenní matici transformace T_H (46).

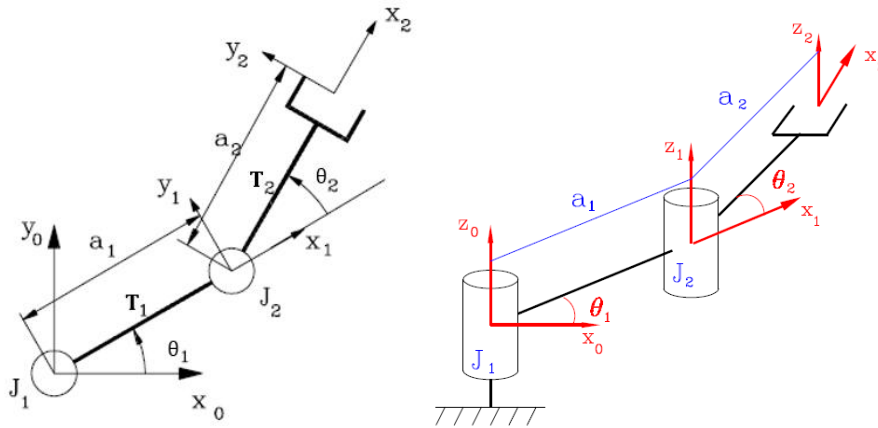
$$T = f(\theta) \quad (46)$$



Obr. 25: Poloha koncového bodu angulárního průmyslového robota T_R pro jednotlivé úhly natočení os θ_1 až θ_6 , obrázek převzat z [33]

Při popisu přímé kinematické úlohy v robotice můžeme postupovat několika způsoby. Na základě znalosti kinematické struktury můžeme sestavit sadu rovnic, které nám vyjádří výslednou polohu efektoru v kartézském souřadnicovém systému, tzn. jednotlivé proměnné T_R . Sestavení rovnic svépomocí pro složitější kinematické struktury může být náročné. Bavíme se totiž o nelineárních vztazích, které bychom bez určité systematickosti nemuseli být schopni nalézt. Druhou možností je využít určitých sestavovacích postupů, které můžeme velmi dobře programově realizovat.

Nejznámějším a nejpoužívanějším postupem sestavení rovnic přímé kinematické úlohy v robotice je bezesporu Denavit-Hartenbergova notace [15], která vychází ze zápisu homogenní matice transformace T_H , a ze které budeme vycházet i v našem případě. Denavit-Hartenbergova notace není jedinou sestavovací technikou. Existují i jiné, např. modifikovaná Denavit-Hartenbergova notace [15], nebo Khalil-Kleinfingerova notace [15]. Jako příklad si sestavíme rovnice přímé kinematické úlohy v robotice dle Denavit-Hartenbergovy notace pro planární manipulátor z obrázku 26.



Obr. 26: Kinematická struktura planárního manipulátoru, obrázek převzat z [12]

Denavit-Hartenbergova notace popisuje transformaci z jedné kinematické vazby do druhé pomocí čtyř parametrů $T_i(\theta, d, a, \vartheta)$, resp. čtyř dílčích transformací, které jsou v následujícím pořadí - rotace kolem osy z $R_z(\theta)$, translace v ose z $P_z(d)$, translace v ose x $P_x(a)$ a rotace kolem osy x $R_x(\vartheta)$ (47, 48). V principu nepotřebujeme žádné další dílčí transformace. Výsledná homogenní matice transformace T_H je pak dána součinem transformací mezi jednotlivými vazbami $T_1 \cdot T_2 \dots T_n$. Souřadný systém je v každé kinematické vazbě, resp. rotačním kloubu J umístěn tak, aby rotace kolem osy z odpovídala úhlu natočení jednotlivých kloubů, tzn. velikosti parametru θ_i . Je nutné poznamenat, že Denavit-Hartenbergova notace dodržuje sekvenci pro sestavení matice rotace $R_z \cdot R_y \cdot R_x$, jelikož R_z předchází R_x , což odpovídá výše předpokládané podobě matice T_H .

$$T_i = R_z \cdot P_z \cdot P_x \cdot R_x \quad (47)$$

$$T_i = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \vartheta & -\sin \vartheta & 0 \\ 0 & \sin \vartheta & \cos \vartheta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (48)$$

Před dosazením do transformačních matic si pro přehlednost sestavíme tabulka Denavit-Hartenbergových parametrů pro kinematickou strukturu planárního manipulátoru z obrázku 26 (viz tabulka 4).

Tab. 4: Tabulka Denavit-Hartenbergových parametrů

Transformace	θ	d	a	ϑ
T_1	θ_1	0	a_1	0°
T_2	θ_2	0	a_2	0°

Výsledná homogenní matice transformace T_H pro kinematickou strukturu planárního manipulátoru je pak vyjádřena vztahem 49, 50 a 51. Můžeme si všimnout, že pro planární manipulátor se dle předpokladu ovlivňují pouze souřadnice x a y na základě hodnoty parametrů θ_1 a θ_2 . Z matice rotace vidíme, že se souřadný systém otáčí opět dle předpokladu pouze podle osy z .

$$T_H = T_1 \cdot T_2 \quad (49)$$

$$T_H = \begin{bmatrix} \cos \theta_1 & -\sin \theta_1 & 0 & a_1 \cos \theta_1 \\ \sin \theta_1 & \cos \theta_1 & 0 & a_1 \sin \theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \theta_2 & -\sin \theta_2 & 0 & a_2 \cos \theta_2 \\ \sin \theta_2 & \cos \theta_2 & 0 & a_2 \sin \theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (50)$$

$$T_H = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) & 0 & a_1 \cdot \cos \theta_1 + a_2 \cdot \cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) & 0 & a_1 \cdot \sin \theta_1 + a_2 \cdot \sin(\theta_1 + \theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (51)$$

Z matice T_H můžeme následně určit translaci a rotaci kartézského souřadného systému, tzn. jednotlivé proměnné T_R (52)

$$\begin{aligned} x &= a_1 \cdot \cos \theta_1 + a_2 \cdot \cos(\theta_1 + \theta_2) \\ y &= a_1 \cdot \sin \theta_1 + a_2 \cdot \sin(\theta_1 + \theta_2) \\ z &= 0 \\ \alpha &= \theta_1 + \theta_2 \\ \beta &= 0 \\ \gamma &= 0 \end{aligned} \quad (52)$$

Po dosazení parametrů θ_1 a θ_2 do rovnic 52 získáme velmi snadno polohu koncové bodu planárního manipulátoru z obrázku 26. Jak již bylo jednou řečeno, pro složitější manipulátory může být hledání těchto rovnic náročné. Při použití Denavit-Hartenbergovy notace se dá sestavování velmi dobře zobecnit a automatizovat.

3.3 Rozměrová analýza

Než se pustíme do řešení inverzní kinematické úlohy v robotice, prodiskutujeme vliv počtu parametrů $\theta_1 \dots \theta_n$, které definují koncovou polohu efektoru na redundanci, resp. neredundaci kinematické struktury. Obecně může být parametrů, které definují koncovou polohu kinematické struktury nekonečně mnoho, přičemž koncová poloha je definovaná šesti parametry T_R (minimální reprezentace). To znamená, že máme k dispozici šest nelineárních rovnic pro n neznámých. Redundance kinematické struktury se zkoumá na základě Jacobiho matice J (54), tzn. na základě matice derivací [19]. Skrze Jacobiho matici totiž definujeme závislost mezi úhlovou rychlostí jednotlivých os $\frac{d\theta}{dt}$ a rychlostí koncového bodu v kartézském souřadnicovém systému $\frac{dT_R}{dt}$ (53). Na základě Jacobiho matice můžeme zkoumat, jak ovlivňuje změna polohy jednotlivých os kinematické struktury polohu koncového bodu. Jacobiho matice v robotice má standardně šest řádků, pokud je rotace popsána třemi parametry (minimální reprezentace), a délku n podle počtu jednotlivých os. Pokud bude $n \leq 6$, jedná se o neredundantní kinematickou strukturu. V případě, že $n > 6$, jedná se o redundantní kinematickou strukturu. V principu se jedná o totéž jako v případě přeúčtené, nebo nedostatečně určené matice soustavy. Redundantní kinematická struktura nabízí nekonečně mnoho řešení inverzní kinematické úlohy v robotice, jelikož máme více neznámých než rovnic. Neredundantní kinematická struktura má zase přesně dané řešení, nebo nemá řešení žádné. Toto tvrzení není ovšem pravidlem, jelikož pro určité postavení kinematické struktury může Jacobiho matice ztratit svou hodnotu, a my pak nebudeme schopni najít pro požadovanou změnu $\frac{dT_R}{dt}$ odpovídající změnu $\frac{d\theta}{dt}$. Takovému postavení kinematické struktury se říká singulární poloha. V singulární poloze pak existuje nekonečně mnoho řešení inverzní kinematické úlohy, nebo neexistuje žádné. V blízkosti singulární polohy musíme také očekávat velké změny rychlosti pro malé změny polohy jednotlivých os. Toto všechno vede na zhoršené schopnosti polohování kinematické struktury v oblasti singularity. Tématu singularity v robotice se věnuje nejrůznější odborná literatura [19-20]. Pro nás je asi nejdůležitější informace ta, že se při výpočtech inverzní kinematické úlohy v robotice používá výše zmiňovaná pseudoinverze. Pseudoinverze nám umožní najít řešení rovnice 53 pro jakoukoliv podobu dynamicky se měnící Jacobiho matice, což je velmi důležitá vlastnost.

$$\frac{dT_R}{dt} = J(\theta) \frac{d\theta}{dt} \quad (53)$$

$$J = \begin{bmatrix} \frac{\partial f_x}{\partial \theta_1} & \frac{\partial f_x}{\partial \theta_2} & \cdots & \frac{\partial f_x}{\partial \theta_n} \\ \frac{\partial f_y}{\partial \theta_1} & \frac{\partial f_y}{\partial \theta_2} & \cdots & \frac{\partial f_y}{\partial \theta_n} \\ \frac{\partial f_z}{\partial \theta_1} & \frac{\partial f_z}{\partial \theta_2} & \cdots & \frac{\partial f_z}{\partial \theta_n} \\ \frac{\partial f_\alpha}{\partial \theta_1} & \frac{\partial f_\alpha}{\partial \theta_2} & \cdots & \frac{\partial f_\alpha}{\partial \theta_n} \\ \frac{\partial f_\beta}{\partial \theta_1} & \frac{\partial f_\beta}{\partial \theta_2} & \cdots & \frac{\partial f_\beta}{\partial \theta_n} \\ \frac{\partial f_\gamma}{\partial \theta_1} & \frac{\partial f_\gamma}{\partial \theta_2} & \cdots & \frac{\partial f_\gamma}{\partial \theta_n} \end{bmatrix} \quad (54)$$

Jako příklad můžeme uvést Jacobiho matici pro planární manipulátor z obrázku 26. Po derivaci rovnic 52 a dosazení do vztahu 54 získáme závislost mezi změnou jednotlivých os a změnou polohy koncového bodu planárního manipulátoru (55). Z Jacobiho matice je vidět, že změna θ_1 a θ_2 má vliv pouze na proměnné x , y a α .

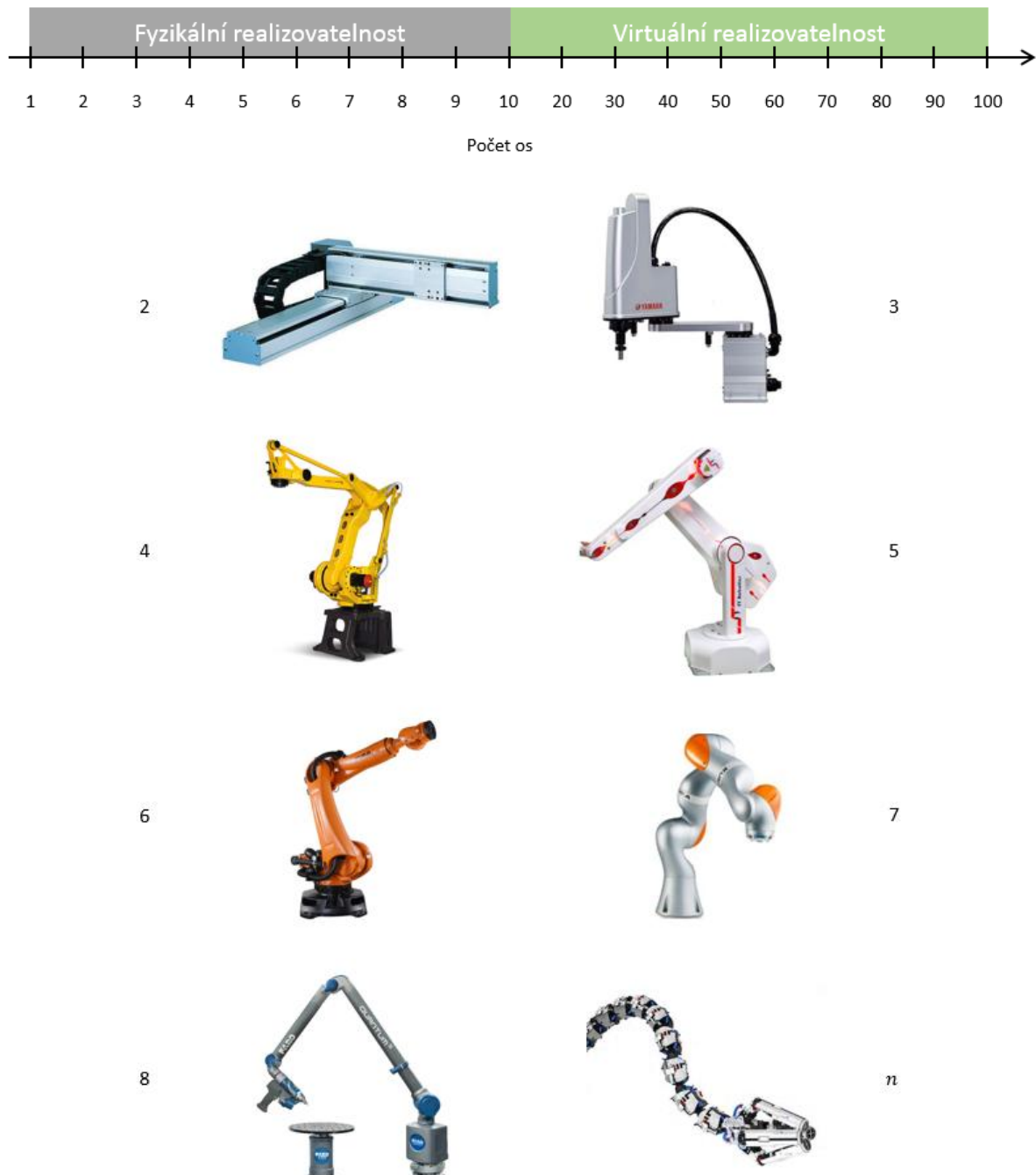
$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} = \begin{bmatrix} -a_2 \cdot \sin(\theta_1 + \theta_2) - a_1 \cdot \sin(\theta_1) & -a_2 \cdot \sin(\theta_1 + \theta_2) \\ a_2 \cdot \cos(\theta_1 + \theta_2) + a_1 \cdot \cos(\theta_1) & a_2 \cdot \cos(\theta_1 + \theta_2) \\ 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \quad (55)$$

Pro případ planárního manipulátoru existuje singulární poloha pro hodnoty $\theta_1 = 0$ a $\theta_2 = 0$. V tomto případě vypadne z Jacobiho matice řádek, který má vliv na změnu proměnné x . Z pozice $\theta_1 = 0$ a $\theta_2 = 0$ tedy nejsme schopni ovlivnit změnu tohoto parametru. Pokud bychom při inverzním výpočtu uvažovali pouze translaci, a zanedbali vazbu pro α , bude hodnota Jacobiho matice rovna jedné, což by znamenalo nekonečně mnoho možných řešení rovnice 56.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ a_1 + a_2 & a_2 \\ 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \quad (56)$$

Pokud připustíme fyzikální realizovatelnost kinematické struktury, není velikost Jacobiho matice omezena jen počtem řádků, ale také počtem sloupců. Pro sériové kinematické struktury můžeme očekávat omezení zhruba na deseti osách. Vycházíme z faktu, že realizace robota pro více jak deset řízených os, je konstrukčně velice obtížný úkol. Pokud bychom porovnali investice do výroby takového robota s jeho praktickým využitím, byl by výsledek značně nerentabilní. Prakticky už je problém uvést příklad robota s osmi, nebo devíti osami, který by se vyráběl v sérové produkci, natož pro robota s deseti a více osami. Naše omezení fyzikální realizovatelnosti je čistě teoretické pro tuto práci a neznamena, že roboty s více osami nelze zkonstruovat. Tímto rozdělením chceme pouze poukázat na fakt, že

v praktických aplikacích, tzn. pro fyzikálně realizovatelné manipulátory, bude mít Jacobiho matice ve většině případech maximálně šest řádků a deset sloupců. Pro takovou velikost matice není časová ztráta výpočtu metody relaxace úhlu, oproti ostatním numerickým metodám až tak výrazná. To nám dává určitou možnost nasazení metody relaxace úhlu právě v oblasti robotiky. Na obrázku 27 je znázorněné pomyslné rozdělení na fyzikálně realizovatelné a virtuálně realizovatelné kinematické struktury. Dále jsou uvedeny některé příklady konkrétních sériových robotů a manipulátorů.



Obr. 27: Příklady sériových robotů a manipulátorů, 2 osy - kartézský manipulátor [34], 3 osy – SCARA robot Yamaha [35], 4 osy – paletizační robot Fanuc [36], 5 os – robot R12 ST Robotics [37], 6 os – robot KR210 R2700 EXTRA KUKA [38], 7 os – LBR IIWA KUKA [39], 8 os – robot FARO Quantum [40], n os – nerealizovaná studie robotického hada [41]

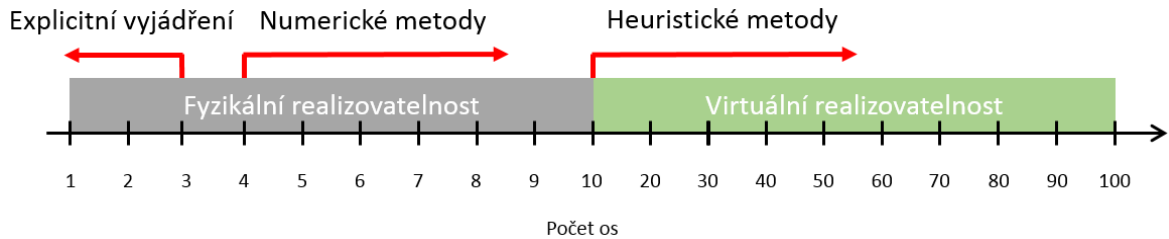
3.4 Inverzní kinematická úloha v robotice

Inverzní kinematická úloha v robotice [9] je zobrazení z prostoru polohy koncového bodu kinematické struktury \mathbf{T} do prostoru kloubových souřadnic $\boldsymbol{\theta}$. To znamená, že známe polohu koncového bodu v kartézském souřadnicovém systému \mathbf{T}_R , resp. homogenní matici transformace \mathbf{T}_H , a hledáme polohy všech rotačních vazeb $\boldsymbol{\theta} = (\theta_0, \theta_1, \dots, \theta_n)$ (viz obr. 24 a rovnice 57).

$$\boldsymbol{\theta} = f^{-1}(\mathbf{T}) \quad (57)$$

Než se pustíme do řešení inverzní kinematické úlohy v robotice pomocí metody relaxace úhlu, uděláme si ještě krátký přehled známých a často používaných metod. Z matematického hlediska znamená řešení inverzní kinematické úlohy v robotice hledání řešení nelineárních rovnic. Přesně těch, které získáme přímou kinematickou úlohou v robotice. Rozdíl je v tom, že poloha rotačních vazeb $\boldsymbol{\theta}$ je pro nás v tomto případě neznámá. Nejjednodušší a nejpřesnější strategií řešení inverzní kinematické úlohy v robotice je upravit nelineární rovnice tak, abychom neznámé mohli explicitně vyjádřit. V odborné literatuře [21] se můžeme setkat s pojmenováním geometrická, nebo také vektorová, metoda. Problém je v tom, že pro složitější nelineární rovnice nebude explicitní vyjádření jednoduché. Takovou pomyslnou hranicí jsou přibližně tři neznámé rotační vazby. Existují také výjimky pro některé speciální kinematické struktury, kde i pro větší počet rotačních vazeb je explicitní vyjádření stále možné. Typickým příkladem je angulární kinematická struktura robota s šesti rotačními vazbami. Kinematickou dekompozicí [19] můžeme rozdělit řešení na dvě samostatné úlohy. Jedna úloha bude pro osy robota 1-3 (manipulátor) a druhá pro osy robota 4-6 (zápěstí). Každá úloha pak bude mít maximálně tři neznámé, tzn. dojde k výraznému ulehčení výpočtu. Další významnou skupinou jsou metody postavené na numerickém řešení nelineárních rovnic. Z obecně známých numerických algoritmů řešení soustav nelineárních rovnic je to např. Newtonova metoda [3], nebo Gradientní metoda [3]. Tak jako Newtonova metoda, tak i další numerické algoritmy v robotice vycházejí ze znalosti Jacobiho matice, např. metoda pseudoinverze Jacobiho matice [20], metoda transponované Jacobiho matice [20], nebo metoda nejmenších čtverců s tlumením, známá jako algoritmus Levenberg-Marquardt [20]. Metody, které jsou postaveny na Jacobiho matici, vycházejí nejčastěji z rovnice 53. Numerické metody použijeme tehdy, pokud nejsme schopni explicitně hledanou polohu rotačních vazeb vyjádřit, tzn. pro čtyři a více neznámých rotačních vazeb, pokud uvážíme naše přibližné omezení pro explicitní vyjádření. Poslední kategorií řešení inverzní kinematické úlohy v robotice jsou metody heuristické, které jsou vhodné jako alternativní řešení problému, pokud jsou klasické numerické metody příliš pomalé, nebo pro nalezení aproximace řešení, pokud ostatní metody selhávají. Jsou to metody postavené většinou na jednoduchém geometrickém principu. Ten však nemusí být funkční pro všechny kinematické struktury. Mezi heuristické metody patří populární algoritmus CCD (Cyclic Coordinate Descent) [27], který se hojně využívá při animaci a v herním průmyslu. Velmi dobrý algoritmus je také FABRIK (forward and backward reaching inverse kinematics) [29], který má dokonce lepší výsledky

než CCD. Je však nutné poznamenat, že neexistuje univerzální metoda. Vždy najdeme výjimky, jako v případě kinematické dekompozice pro angulární kinematickou strukturu. Proto musíme při výběru vhodné metody vycházet ze znalosti kinematické struktury a z požadavků aplikace. Naše rozdělení metod do tří základních skupin je znázorněné na obrázku 28. Toto rozdělení není obecným pravidlem. Slouží pouze pro názornost této práce.



Obr. 28: Metody řešení inverzní kinematické úlohy v robotice – jednoduché rozdělení

Abychom mohli použít metodu relaxace úhlu pro řešení inverzní kinematické úlohy v robotice, zaměříme se na postupy, které vychází ze znalosti Jacobiho matice, tzn. na Newtonovu metodu, metodu pseudoinverze Jacobiho matice a algoritmus Levenberg-Marquardt. Vybrané metody upravíme do podoby soustavy lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, aby byly naším novým algoritmem relaxace úhlu řešitelné.

Iterační předpis pro Newtonovu metodu s pseudoinverzí Jacobiho matice $\mathbf{J}(\boldsymbol{\theta}_k)^+$ je popsán vztahem 58. Obecný princip Newtonovy metody je uveden např. zde [3]. První krok Newtonovy metody vychází z počátečního odhadu $\boldsymbol{\theta}$. Abychom vlivem nesprávného počátečního odhadu $\boldsymbol{\theta}$ nekonvergovali k nějakému lokálnímu minimu, tzn. ke špatnému řešení, doporučuje se volit $\boldsymbol{\theta}$ na základě známého postavení kinematické struktury, které bude blízké žádané hodnotě. Obecně pro numerické metody platí, že při řešení inverzní kinematické úlohy v robotice vycházíme ze známé pozice kinematické struktury a hledáme řešení v nějakém blízkém okolí. Pro velké změny žádané hodnoty nemusí iterační předpis kvůli nelinearitě rovnic fungovat. Toto se v praxi řeší rozdělením změny žádané hodnoty do několika menších kroků, které nás postupně dovedou k cíli. Úprava iteračního předpisu Newtonovy metody do podoby soustavy lineárních rovnic $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ je popsána vztahem 59.

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \mathbf{J}(\boldsymbol{\theta}_k)^+ \cdot \mathbf{T}(\boldsymbol{\theta}_k) \quad (58)$$

$$\mathbf{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{\theta}_{k+1} = \mathbf{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{\theta}_k + \mathbf{T}(\boldsymbol{\theta}_k) \quad (59)$$

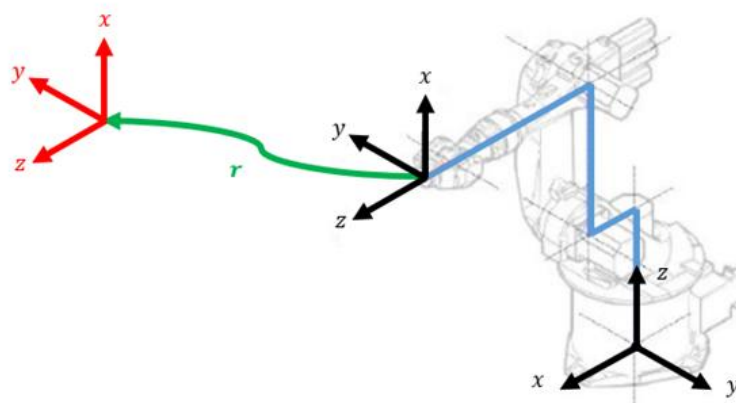
Metoda pseudoinverze Jacobiho matice vychází z již výše uvedené rovnice 53. Po její linearizaci v nějakém malém okolí koncového bodu kinematické struktury získáme rovnici 60. Změna koncového bodu $\Delta \mathbf{T}$ mezi aktuální a žádanou hodnotou je vlastně reziduum \mathbf{r} , které chceme minimalizovat (viz obr. 29). Chceme najít takovou změnu $\Delta \boldsymbol{\theta}$, kterou když přičteme k aktuální hodnotě $\boldsymbol{\theta}$, tak dosáhneme požadovaného cíle. Tomuto principu odpovídá iterační předpis metody

pseudoinverze Jacobiho matice 61. Často se uvádí v podobě, která odpovídá řešení pomocí metody nejmenších čtverců, tzn. rovnice 61 je ještě vynásobena z obou stran $J^T(\theta_k)$. To pak vede na iterační předpis 62.

$$J(\theta) \cdot \Delta\theta = \Delta T = r \quad (60)$$

$$\theta_{k+1} = \theta_k + J(\theta_k)^+ \cdot r_k \quad (61)$$

$$\theta_{k+1} = \theta_k + J^T(\theta_k) \cdot (J(\theta_k) \cdot J^T(\theta_k))^+ \cdot r_k \quad (62)$$



Obr. 29: Rozdíl žádané a aktuální polohy koncového bodu kinematické struktury. Pro velké r je nutné rozdělit dráhu na několik menších úseků kvůli nelinearitě, obrázek převzat z [29]

Opět můžeme vztah 61 upravit do podoby soustavy lineárních rovnic $A \cdot x = b$ (viz iterační předpis 63). Je nutné konstatovat, že pro potřeby řešení pomocí metody relaxe úhlu je tento vztah daleko výhodnější, než předpis vycházející z metody nejmenších čtverců (62). Metoda nejmenších čtverců zbytečně zhoršuje výslednou podmíněnost matice soustavy. Metoda nejmenší čtverců je výhodná pro nasazení pseudoinverze, jelikož získáváme potřebnou symetrickou pozitivně definitní matici již v základním iteračním předpisu $J(\theta_k) \cdot J^T(\theta_k)$. Budeme proto preferovat čistou lineární závislost, která z hlediska podmíněnosti matice povede k lepším časovým výsledkům.

$$\begin{aligned} J(\theta_k) \cdot \Delta\theta_k &= r_k \\ \theta_{k+1} &= \theta_k + \Delta\theta_k \end{aligned} \quad (63)$$

Třetí a poslední metodu, kterou upravíme, bude algoritmus Levenberg-Marquardt (64). Nejedná se o nic jiného, než o metodu nejmenších čtverců s tlumením. Iterační předpis 62 je navíc vybaven konstantou λ , která se v robotice používá pro omezení velkých skokových změn v oblasti singularity. Později si ukážeme její vliv na průběh sledování žádané hodnoty koncového bodu kinematické struktury.

$$\theta_{k+1} = \theta_k + J^T(\theta_k) \cdot (J(\theta_k) \cdot J^T(\theta_k) + \lambda^2 \cdot I)^+ \cdot r_k \quad (64)$$

Při úpravě vztahu 64 do podoby soustavy lineárních rovnic se budeme opět snažit obejít zbytečné navýšení čísla podmíněnosti výsledné matice soustavy. Metodu nejmenších čtverců zredukujeme na lineární závislost, ale konstantu tlumení λ ponecháme. Ve výsledku se nejedná o nic jiného, než o rozšíření soustavy o několik dalších rovnic (65). Zapracování konstanty tlumení nás bude stát nějaký iterační čas navíc, jelikož výsledná soustava lineárních rovnic bude o něco větší.

$$\begin{aligned} \begin{pmatrix} \mathbf{J}(\boldsymbol{\theta}_k) \\ \lambda^2 \cdot \mathbf{I} \end{pmatrix} \cdot \Delta\boldsymbol{\theta}_k &= \begin{pmatrix} \mathbf{r}_k \\ \mathbf{0} \end{pmatrix} \\ \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \Delta\boldsymbol{\theta}_k \end{aligned} \quad (65)$$

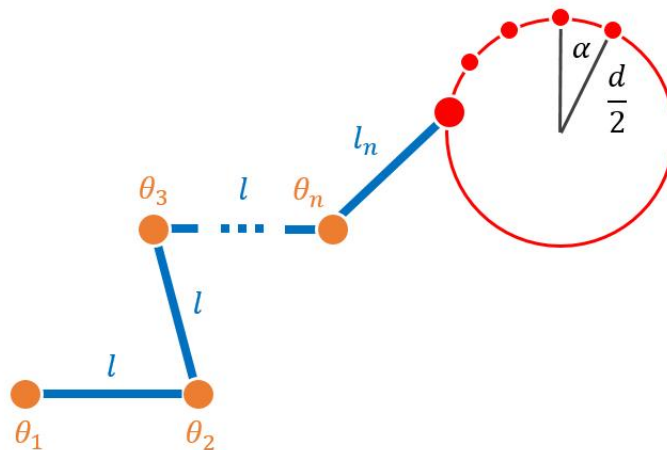
3.5 Porovnání vybraných numerických metod

Teď už nám nezbyvá nic jiného, než výše popsané numerické metody řešení inverzní kinematické úlohy v robotice odzkoušet a vzájemně porovnat. Bude nás zajímat schopnost algoritmu sledovat předepsanou trajektorii. V našem případě to budou dvě kružnice vykreslené koncovým bodem kinematické struktury. Z toho jedna bude procházet singulární polohou. Parametry algoritmu nastavíme tak, aby maximálně odpovídaly požadavkům reálného řízení. Porovnání numerických metod provedeme pro n planárních manipulátorů a pro průmyslového robota KUKA KR210 R2700 EXTRA, který je nejčastějším typem nasazeným v rámci svařoven firmy ŠKODA AUTO a.s. U standardních průmyslových robotů se opakovatelná přesnost pohybuje v setinách milimetrů. U robota KUKA KR210 R2700 EXTRA je to $\pm 0,06$ mm. V našem testu převezmeme tuto hodnotu jako podmínku zastavení numerické metody, tzn. $\max|\mathbf{r}_k| < 0,06$. Co se týká časových nároků, roboty KUKA aktualizují požadovanou hodnotu polohy v IPO (Input-Process-Output) taktu 12 ms. Budeme proto požadovat, aby bylo dosaženo předepsané přesnosti v tomto čase.

Kinematická struktura planárního manipulátoru bude složena z n rotačních vazeb a n spojů (viz obr. 30). Délka jednoho kinematického spoje l bude pro zjednodušení vždy stejná. Koncovým bodem planárního manipulátoru budeme vykreslovat kružnici s průměrem d a úhlovým krokem α . V prvním případě bude kružnice začínat v náhodném postavení mimo singulární polohu planárního manipulátoru. V druhém případě bude kružnice začínat v singulární poloze planárního manipulátoru, tzn. pro $\boldsymbol{\theta} = \mathbf{0}$, a zase v ní končit. Postupně budeme měnit parametry planárního manipulátoru a měřit čas potřebný pro dosažení požadované přesnosti u jednotlivých numerických metod. Velikost planárního manipulátoru bude maximálně $n = 10$, tzn. velikost bude omezena námi zvolenou mezí fyzikální realizovatelnosti. Orientace koncového bodu nebude předepsaná. Trajektorie bude definovaná pouze polohou v rovině x a y .

Co se týká velikosti Jacobiho matice, ta bude mít v případě planárního manipulátoru dva řádky a n sloupců podle zvoleného počtu rotačních vazeb. V části věnované rozměrové analýze jsme definovali Jacobiho matici jako matici derivací. Někdy se také setkáme s pojmem analytická Jacobiho matice. Při jejím sestavování narazíme často na problém s derivací poměrně složitých vztahů. Tomu se v robotice vyhýbáme použitím tzv. geometrické Jacobiho matice [29]. Geometrická Jacobiho matice nahrazuje derivace jednodušší geometrickou formou (66). Vektor \mathbf{z}_i je směr osy rotace jednotlivých rotačních vazeb aktuálního postavení kinematické struktury. Tento vektor odpovídá třetímu sloupci homogenní matice transformace. Vektor \mathbf{P}_i odpovídá čtvrtému sloupci homogenní matice transformace (viz. obr 23). Pozn.: poslední řádek homogenní matice transformace se nepočítá. \mathbf{P}_n je pak poloha koncového bodu kinematické struktury. Jednotlivé vektory získáváme na základě Denavit–Hartenbergovy notace, tzn. \mathbf{z}_0 a \mathbf{P}_0 z matice \mathbf{B}_H , \mathbf{z}_1 a \mathbf{P}_1 z matice \mathbf{T}_1 , \mathbf{z}_2 a \mathbf{P}_2 z matice $\mathbf{T}_1 \cdot \mathbf{T}_2$ atd. Tímto způsobem velice elegantně obejdeme možné problémy při hledání složitých derivací.

$$\mathbf{J} = \begin{bmatrix} \mathbf{z}_0 \times (\mathbf{P}_n - \mathbf{P}_0) & \mathbf{z}_1 \times (\mathbf{P}_n - \mathbf{P}_1) & \dots & \mathbf{z}_0 \times (\mathbf{P}_n - \mathbf{P}_{n-1}) \\ \mathbf{z}_0 & \mathbf{z}_1 & \dots & \mathbf{z}_{n-1} \end{bmatrix} \quad (66)$$



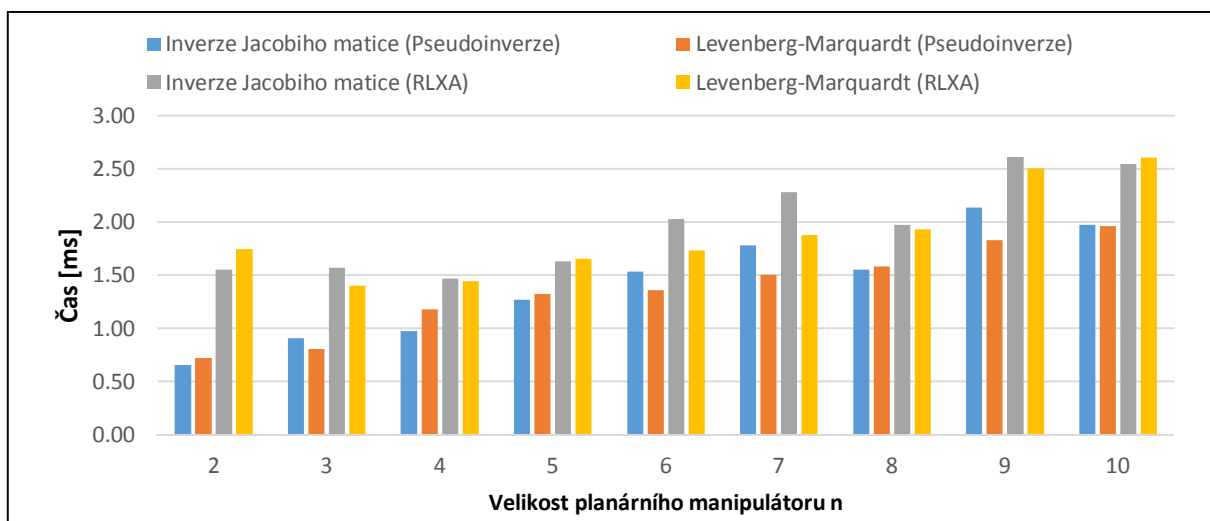
Obr. 30: Planární manipulátor s n rotačními vazbami a n spoji obkreslující kružnici s úhlovým krokem α

Tab. 5: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice planárního manipulátoru z obrázku 30

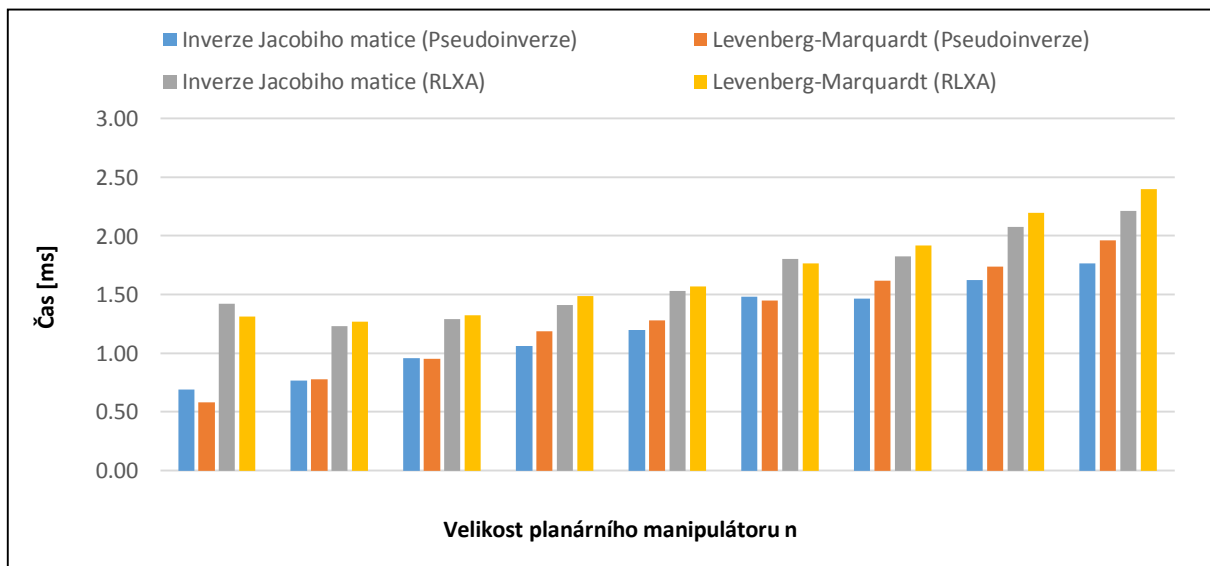
Úhlový krok α	Velikost planárního manipulátoru n	Pseudoinverze						Metoda relaxace úhlu					
		Bez průchodu singularitou			S průchodem singularitou			Bez průchodu singularitou			S průchodem singularitou		
		Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt	Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt	Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt	Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt
\emptyset Čas metody potřebný pro výpočet jednoho úhlového kroku [ms] s požadovanou přesností ± 0.06 mm													
$\pi/25$	2	0.67	0.66	0.72	0.67	1.29	0.69	5.92	1.56	1.75	10.88	7.21	5.17
	3	0.84	0.91	0.81	0.84	1.49	0.89	13.49	1.57	1.40	587.38	71.92	47.67
	4	0.92	0.98	1.18	1.06	1.84	1.07	36.59	1.47	1.44	14.24	10.79	6.32
	5	1.05	1.27	1.32	1.19	1.44	1.27	92.34	1.63	1.66	682.40	72.46	60.65
	6	1.18	1.53	1.36	1.42	2.27	1.41	124.84	2.03	1.74	26.04	15.15	13.22
	7	1.29	1.79	1.51	1.55	2.47	1.63	269.96	2.28	1.88	790.10	99.81	77.05
	8	1.41	1.55	1.58	1.77	2.51	1.83	263.23	1.97	1.93	42.18	20.21	15.91
	9	1.55	2.14	1.83	1.81	3.01	2.06	279.92	2.61	2.51	934.92	130.13	99.14
	10	1.69	1.97	1.96	1.96	2.90	2.31	296.11	2.55	2.61	83.60	44.48	20.31
	$\pi/50$	2	0.58	0.70	0.59	0.57	0.79	0.63	4.67	1.42	1.31	8.74	5.26
3		0.77	0.77	0.78	0.74	1.03	0.84	13.20	1.23	1.27	671.35	59.03	30.91
4		0.87	0.96	0.95	0.92	1.36	0.88	30.61	1.29	1.33	13.37	9.86	4.31
5		1.03	1.06	1.19	0.99	1.54	1.06	65.85	1.41	1.49	731.46	68.48	39.78
6		1.13	1.20	1.28	1.18	1.75	1.31	121.22	1.53	1.57	20.44	17.01	11.27
7		1.25	1.48	1.45	1.26	1.95	1.36	271.91	1.80	1.77	887.58	83.11	45.17
8		1.41	1.47	1.62	1.45	1.96	1.61	250.27	1.83	1.92	34.76	17.44	15.48
9		1.51	1.63	1.74	1.54	2.39	1.73	316.35	2.08	2.20	948.94	104.68	59.00
10		1.68	1.77	1.96	1.87	2.42	1.90	287.40	2.22	2.40	67.98	20.02	16.04
$\pi/100$		2	0.56	0.59	0.94	0.55	0.57	0.55	4.52	1.05	1.66	8.56	1.96
	3	0.81	0.72	1.03	0.70	0.73	0.76	13.34	1.05	1.31	660.54	18.68	19.30
	4	0.90	0.87	1.08	0.85	0.88	0.90	33.92	1.16	1.40	11.97	4.17	3.27
	5	1.17	0.98	1.11	0.98	0.97	1.05	90.54	1.25	1.35	737.34	21.79	24.04
	6	1.12	1.15	1.38	1.11	1.18	1.24	119.35	1.41	1.58	24.38	6.35	4.92
	7	1.24	1.31	1.50	1.25	1.28	1.37	272.50	1.60	1.73	854.17	26.55	28.35
	8	1.35	1.45	1.58	1.41	1.39	1.55	254.32	1.77	1.82	34.17	6.94	7.58
	9	1.51	1.61	1.78	1.52	1.54	1.69	272.51	1.97	2.13	954.95	31.74	36.52
	10	1.62	1.71	1.89	1.65	1.67	1.81	263.78	2.12	2.28	82.59	9.30	16.13

Výsledky pro experiment s planárním manipulátorem z obrázku 30 jsou uvedeny v tabulce 5. Zde jsou zvýrazněné hodnoty, které nespĺnily výše uvedené kritérium řešení do 12 ms. Na obrázku 31, 32 a 33 jsou ještě další grafická porovnání. Můžeme konstatovat, že pokud se manipulátor nachází mimo singulární polohu, je metoda inverze Jacobiho matice a metoda Levenberg-Marquardt postavená na metodě relaxace úhlu naprosto srovnatelná s výsledky pseudoinverze. Newtonova metoda se ukázala jako nevhodná bez ohledu na postavení manipulátoru. V případě průchodu manipulátoru singulární polohou nejsou výsledky metody relaxace úhlu tak dobré jako v případě pseudoinverze. To je pochopitelné, jelikož singulární poloha manipulátoru znamená práci se špatně podmíněnou maticí soustavy, a to není pro metodu relaxace úhlu příliš vhodné. Každopádně je singulární postavení manipulátoru extrém, který se v praxi často hlídá. Je pravidlem, že se při tvorbě trajektorie singulární poloze vyhýbáme. Je běžné, že řídicí systémy ani takovou trajektorii nedovolí vytvořit. Pokud budeme

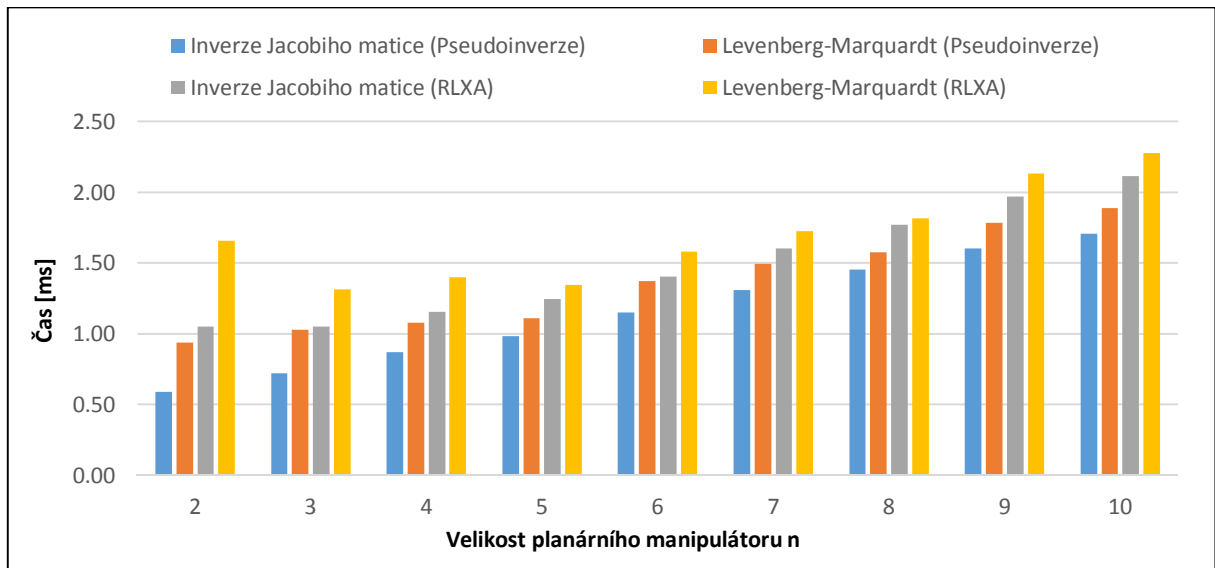
při nasazení metody relaxace úhlu uvažovat stejným způsobem, nemusí nás zhoršené časové výsledky v singulární poloze až tak zajímat. Pokud se však nemůžeme singulární poloze vyhnout, můžeme zkusit zvýšené výpočetní nároky eliminovat. První věc, která bezesporu pomůže, je zpomalit robota a ponechat více času na plánování trajektorie. Tím sice nezrychlíme čas výpočtu, ale v konečné instanci budou dosažené výsledky více přijatelné. Také si můžeme všimnout, že při zmenšování úhlového kroku se jednotlivé výsledky postupně zlepšují. Dále můžeme hledat vhodné nastavení konstanty tlumení u metody Levenberg-Marquardt (viz obr. 34). Pokud nám to situace dovolí, můžeme pracovat s menší požadovanou přesností. V neposlední řadě můžeme zařadit do výpočtu předpodmiňování matice soustavy. Pokud se budeme držet některého z těchto postupů, můžeme v oblasti singularity časové nároky metody relaxace úhlu výrazně zlepšit.



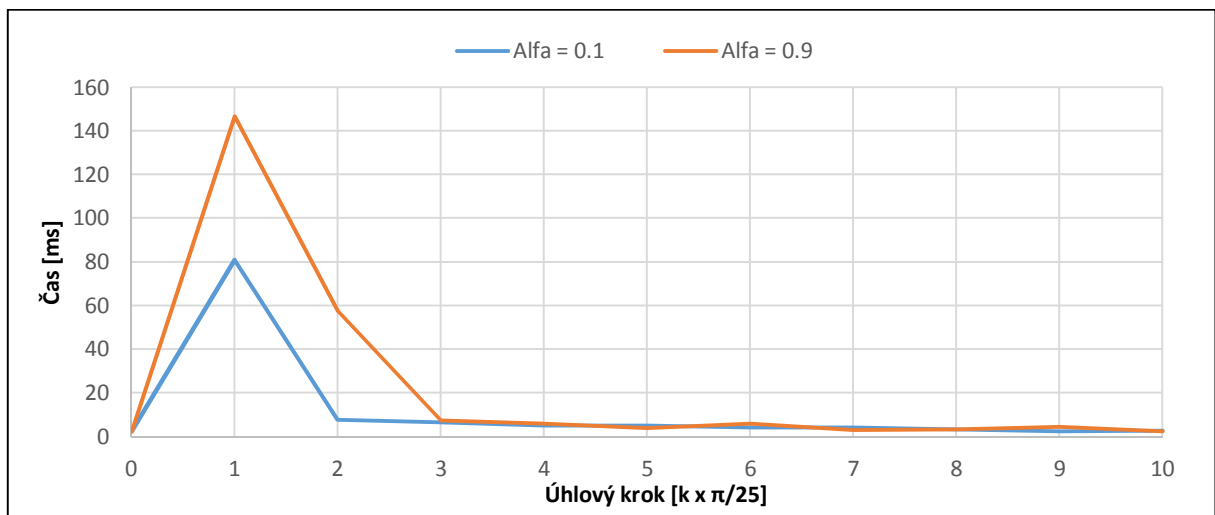
Obr. 31: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/25$



Obr. 32: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/50$

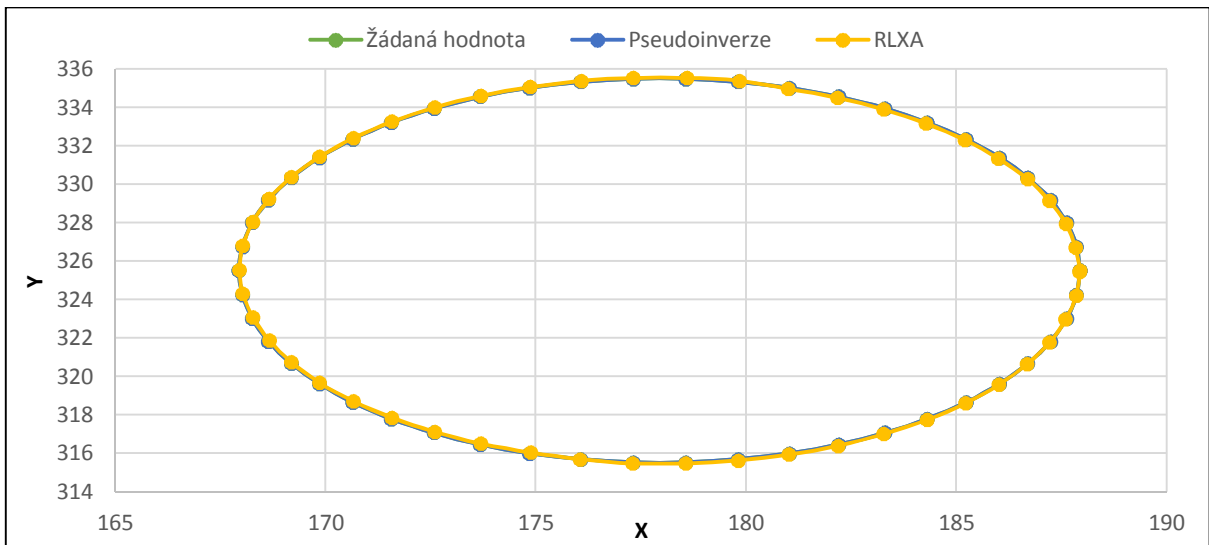


Obr. 33: Časové porovnání vybraných numerických metod mimo singulární polohu s úhlovým krokem $\pi/100$

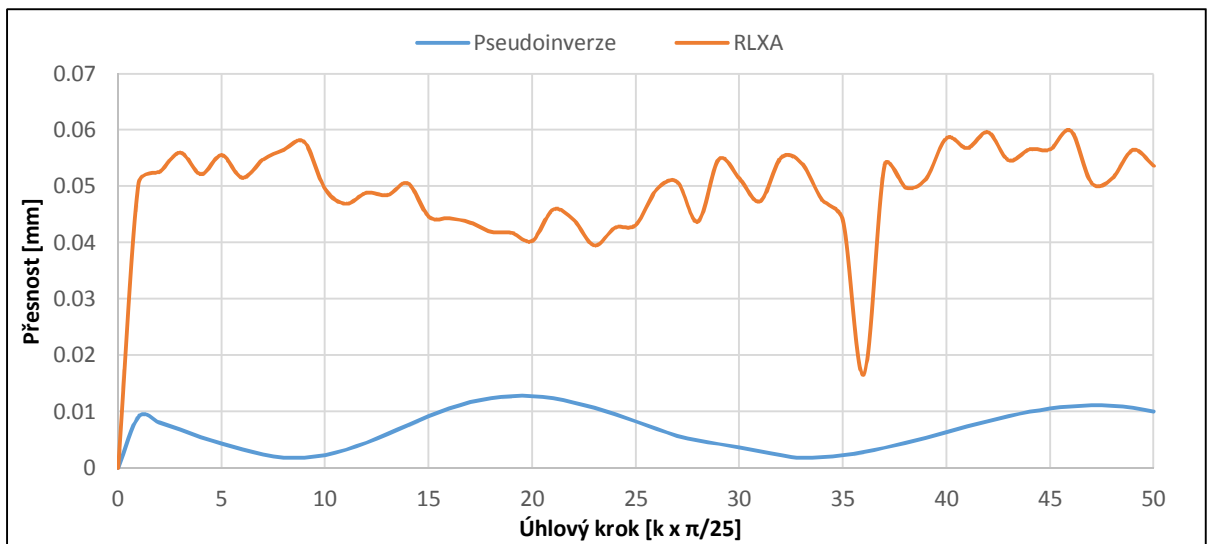


Obr. 34: Čas výpočtu při průchodu singulární polohou, planární manipulátor $n = 2$, metoda Levenberg-Marquardt s koeficientem tlumení alfa

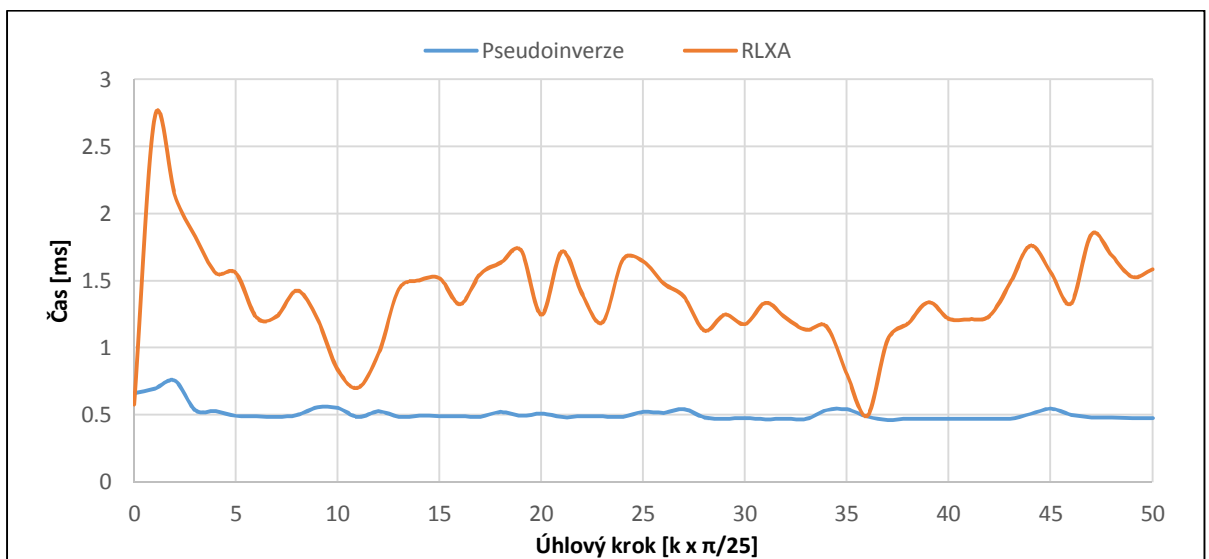
Na obrázku 35, 36 a 37 je ukázka průběhu řešení inverzní kinematické úlohy v robotice pomocí inverze Jacobiho matice pro planární manipulátor s počátečním postavením mimo singulární polohu s konfigurací $n = 2$ a úhlovým krokem $\pi/25$. Planární manipulátor obkresluje kružnici s poloměrem 10 mm. Na obrázku 35 je vidět, jak vybrané metody sledují průběh žádané hodnoty. Není zde vidět téměř žádný rozdíl, jelikož výsledná přesnost jednoho úhlového kroku je vždy menší než 0,06 mm. Detail průběhu přesnosti je vidět na obrázku 36. Metoda relaxace úhlu se pohybuje s přesností těsně pod hranicí 0,06 mm, jelikož se výpočet při dosažení této hranice okamžitě ukončuje. V případě pseudoinverze je výsledná přesnost nižší. Rozdíl je pouze v nastavení počtu iterací u metody relaxace úhlu v rámci jednoho iteračního kroku inverze Jacobiho matice. Obě dvě metody se zastavují stejnou přesností. V rámci metody relaxace úhlu se ovšem pracuje s jemnějším krokem, a proto lze zastavit výpočet dříve.



Obr. 35: Sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice



Obr. 36: Přesnost při sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice



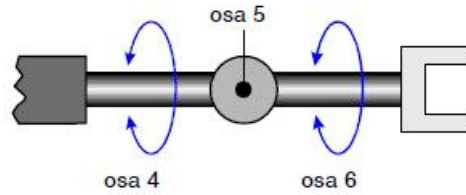
Obr. 37: Čas výpočtu při sledování žádané hodnoty, planární manipulátor $n = 2$, metoda inverze Jacobiho matice

Na základě měření času výpočtu potřebného pro dosažení přesnosti 0,06 mm není mimo oblast singularity téměř žádný rozdíl mezi použitím pseudoinverze a metody relaxace úhlu (viz obr. 36, 37). Rozdíl se pohybuje v řádu jednotek milisekund. Je také nutné konstatovat, že porovnáváme optimalizovanou MATLAB knihovnu pro výpočet pseudoinverze s ukázkovým kódem výpočtu metody relaxace úhlu v maticové podobě viz obr. 10. Rozhodně je zde prostor pro další funkční odladění výpočtu. Už jsme zde uváděli, že výpočet v maticové podobě zbytečně počítá hodnoty pod a nad diagonálou znaménkové matice. Dále můžeme optimalizovat okamžité zastavení algoritmu při dosažení předepsané přesnosti. Na obr. 36 je vidět, že žádanou přesnost zbytečně překročíme.

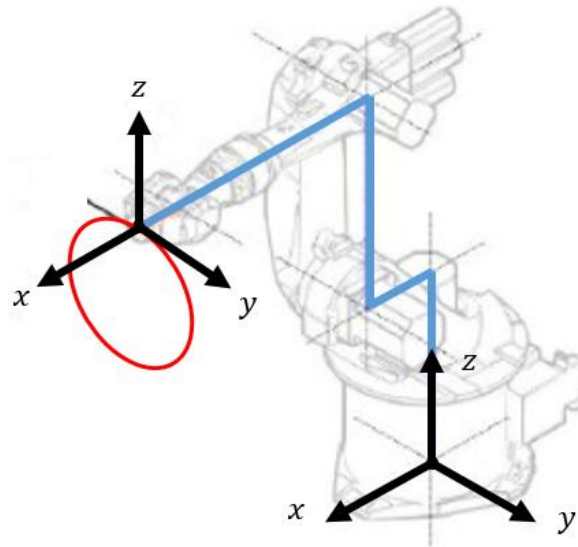
Z teoretické roviny planárního manipulátoru se nyní pustíme do řešení praktické úlohy s průmyslovým robotem KUKA KR210 R2700 EXTRA. Nejprve je nutné sestavit transformace mezi jednotlivými kinematickými vazbami robota na základě Denavit–Hartenbergovy notace. V tabulce 6 je uveden přehled Denavit–Hartenbergových parametrů pro robota KUKA KR210 R2700 EXTRA. Kvůli rozsahu zde jednotlivé transformace uvádět nebudeme. Stačí dosadit parametry z tabulky 5 do vztahu 46. Ze známé transformace se následně určí Jacobiho matice na základě vztahu 65. Stejně jako v případě planárního manipulátoru necháme robota vykreslovat kružnici v rovině y a z o poloměru 10 mm koncovým bodem kinematické struktury (viz obr. 39). Parametr polohy x a parametry rotace α , β , γ budou konstantní, ale ve výpočtu budou figurovat jako požadované. Sledování žádané hodnoty provedeme metodou inverze Jacobiho matice a metodou Levenberg-Marquardt. Newtonovu metodu zkoušet nebudeme, jelikož výsledky s planárním manipulátorem ukázaly, že tato metoda nevyhází dobře v kombinaci s metodou relaxace úhlu. Opět budeme měřit čas potřebný pro dosažení žádané hodnoty jednoho úhlového kroku s přesností 0,06 mm při použití pseudoinverze a metody relaxace úhlu. Měření provedeme pro základní postavení robota mimo singulární polohu a pro základní postavení robota v singulární poloze (viz obr. 38, 39).

Tab. 6: Denavit-Hartenbergovy parametry robota KUKA KR210 R2700 EXTRA

Transformace	θ	d	a	ϑ
T_1	θ_1	675	350	-90°
T_2	$\theta_2 - 90^\circ$	0	1150	0°
T_3	θ_3	0	-41	90°
T_4	θ_4	-1200	0	-90°
T_5	θ_5	0	0	90°
T_6	$\theta_6 + 180^\circ$	-240	0	180°



Obr. 38: Singulární poloha osy A5 robota KUKA KR210 R2700 EXTRA (počáteční bod), obrázek převzat z [42]

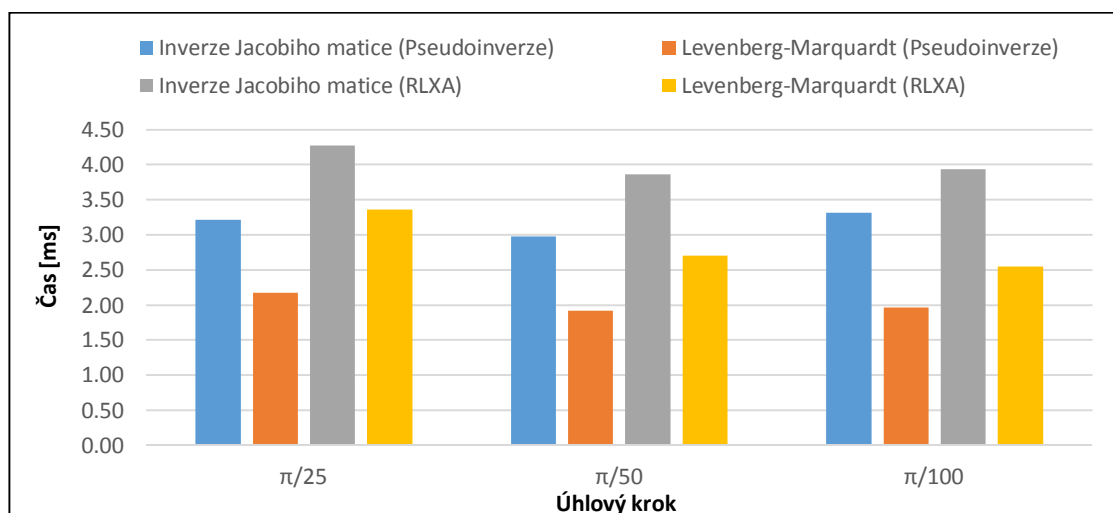


Obr. 39: Robot KUKA KR210 R2700 EXTRA obkreslující kružnici v rovině z a y, obrázek převzat z [29]

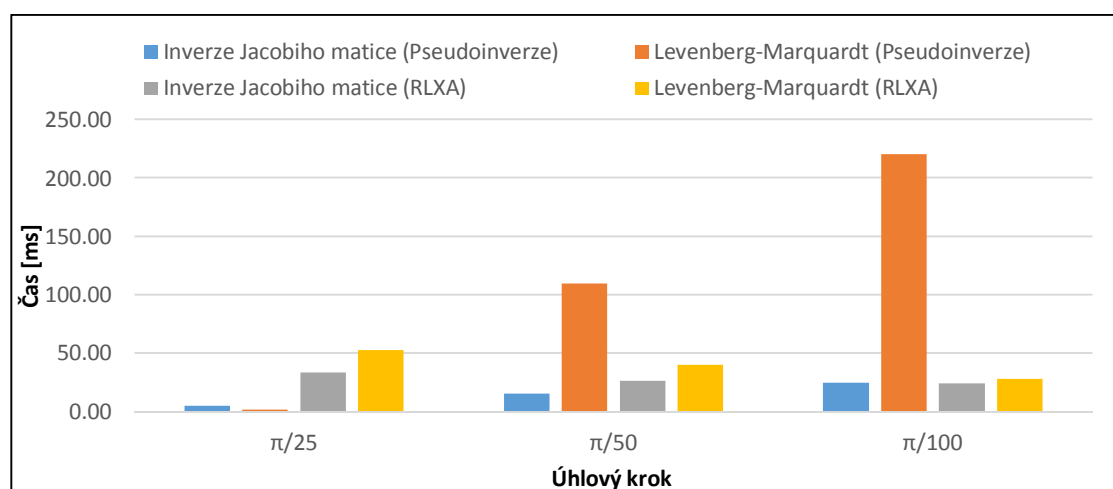
V tabulce 6 jsou uvedeny všechny naměřené hodnoty z našeho experimentu s robotem KUKA KR210 R2700 EXTRA. Je nutné konstatovat, že doba výpočtu metody relaxace úhlu je mimo singulární polohu robota plně srovnatelná s časem dosaženým při použití pseudoinverze. Další grafické srovnání průměrného času výpočtu jednoho úhlového kroku mimo singulární polohu je na obrázku 40. V oblasti singulární polohy lze pozorovat zpomalení výpočtu u obou metod (viz tab. 7 a obr. 41). Obě dvě metody měly problém s dosažením času 12 ms při zachování požadované přesnosti 0.06 mm. Tady je nutné upozornit na zajímavé chování naměřených výsledků v závislosti na zmenšování úhlového kroku. Při zmenšování úhlového kroku se čas výpočtu u metody relaxace úhlu zlepšuje a u pseudoinverze výrazně zhoršuje. Nejvíce patrné je to u metody Levenberg-Marquardt. Tohoto efektu jsme si všimli už při výpočtu planárního manipulátoru s tím rozdílem, že u planárního manipulátoru čas řešení pomocí pseudoinverze výrazně nerostl. Důvod tohoto chování se pokusíme později objasnit. Co se týká eliminace nárůstu času v oblasti singulární polohy, opět zde platí stejná pravidla jako v případě planárního manipulátoru. Nejlepší postup v rámci plánování trajektorie je maximálně se singulární poloze vyhnout.

Tab. 7: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice průmyslového robota KUKA R210 R2700 EXTRA, poloměr kružnice 10 mm

Úhlový krok α	Pseudoinverze				Metoda relaxace úhlu			
	Bez průchodu singularitou		S průchodem singularitou		Bez průchodu singularitou		S průchodem singularitou	
	Inverze Jacobiho matice	Levenberg-Marquardt	Inverze Jacobiho matice	Levenberg-Marquardt	Inverze Jacobiho matice	Levenberg-Marquardt	Inverze Jacobiho matice	Levenberg-Marquardt
	Ø Čas metody potřebný pro výpočet jednoho úhlového kroku [ms] s požadovanou přesností ± 0.06 mm							
$\pi/25$	3.22	2.18	5.06	2.13	4.28	3.36	33.94	52.90
$\pi/50$	2.98	1.92	15.52	109.61	3.87	2.71	26.81	40.20
$\pi/100$	3.32	1.97	24.77	220.45	3.94	2.55	24.45	28.52

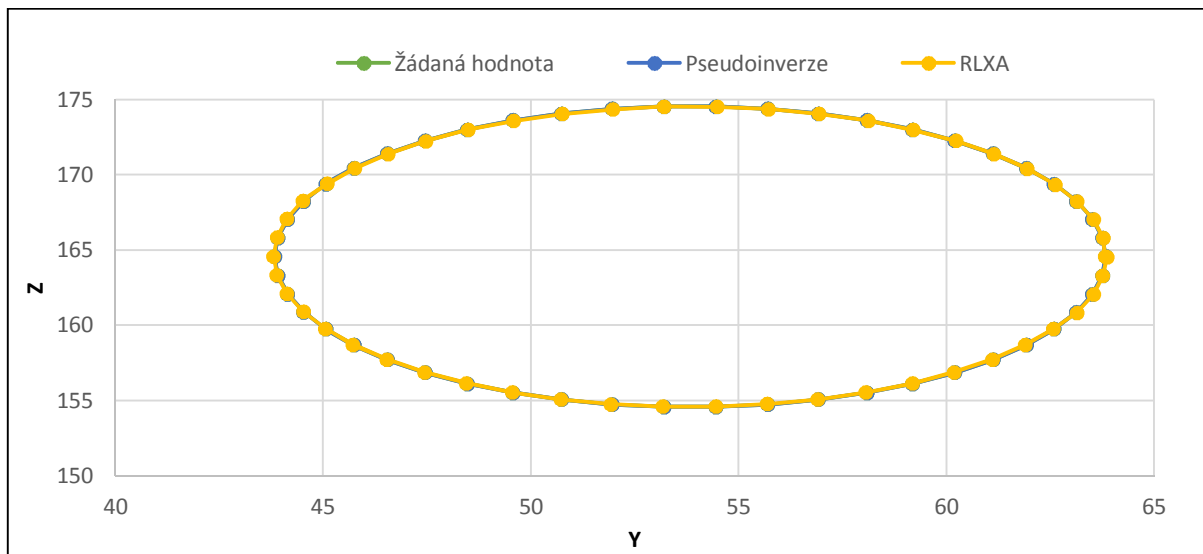


Obr. 40: Čas výpočtu jednoho úhlového kroku robota KUKA KR210 R2700 EXTRA bez průchodu singulární polohou

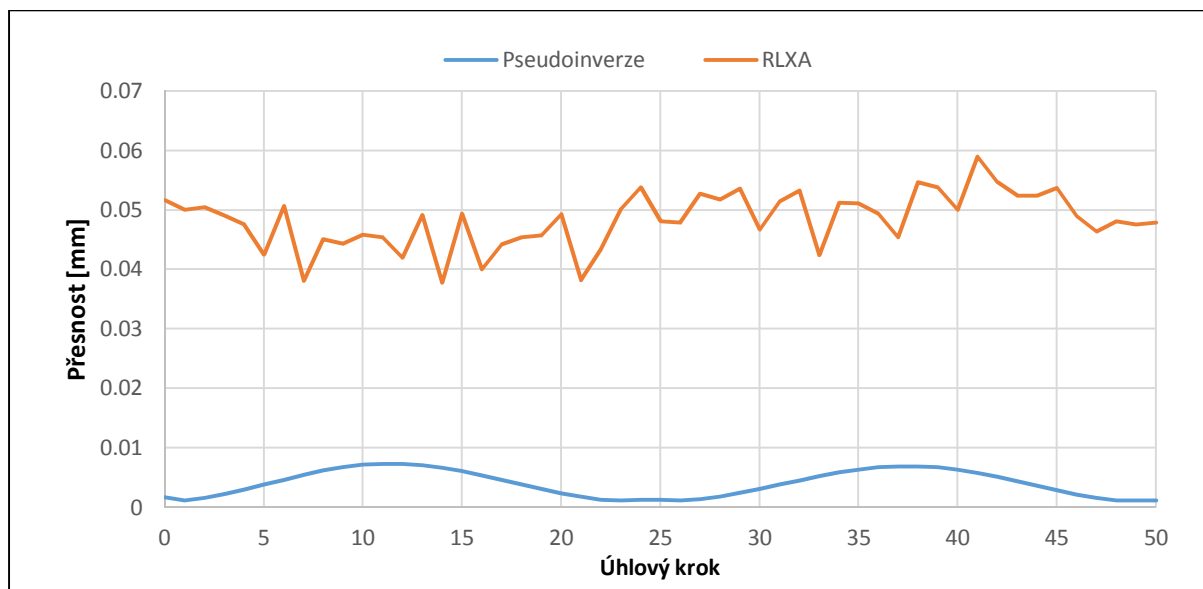


Obr. 41: Čas výpočtu jednoho úhlového kroku robota KUKA KR210 R2700 EXTRA s průchodem singulární polohou

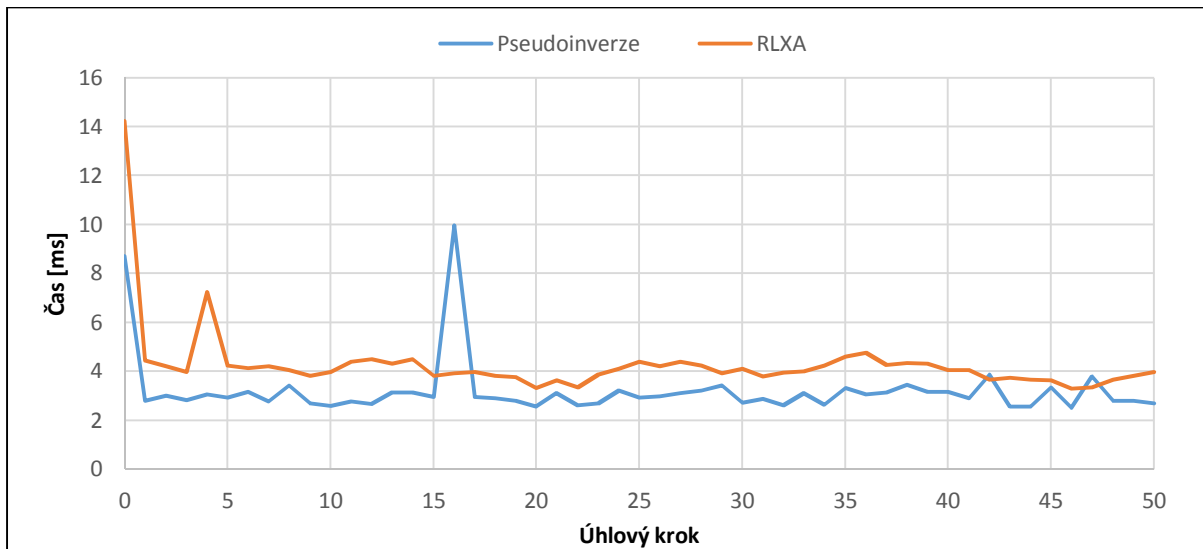
Na obrázku 42, 43 a 44 je ukázka průběhu řešení inverzní kinematické úlohy v robotice průmyslového robota KUKA KR210 R2700 EXTRA. Jedná se o algoritmus inverze Jacobiho matice v náhodném počátečním postavením mimo singulární polohu s úhlovým krokem $\pi/25$. Robot vykresluje kružnici s poloměrem 10 mm v rovině y a z. Na průběhu přesnosti (viz obr. 43) můžeme opět pozorovat zastavení výpočtu při překročení hranice 0,06 mm. Na obrázku 44 si můžeme všimnout, že ne v každém bodě je výpočet pomocí pseudoinverze rychlejší. Někdy i výpočet pomocí pseudoinverze potřebuje více času, než metoda relaxace úhlu. Obdobné průběhy jsme získali v jakémkoliv jiném místě mimo oblast singulárního postavení robota.



Obr. 42: Sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu s úhlovým krokem $\pi/25$, inverze Jacobiho matice

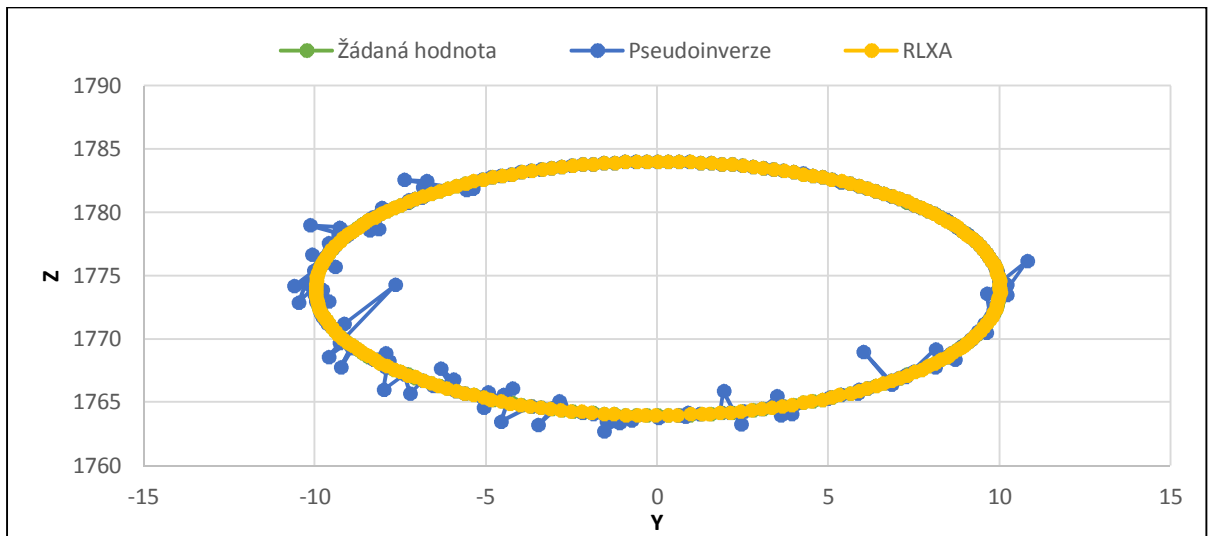


Obr. 43: Přesnost jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu, úhlový krok $\pi/25$, inverze Jacobiho matice

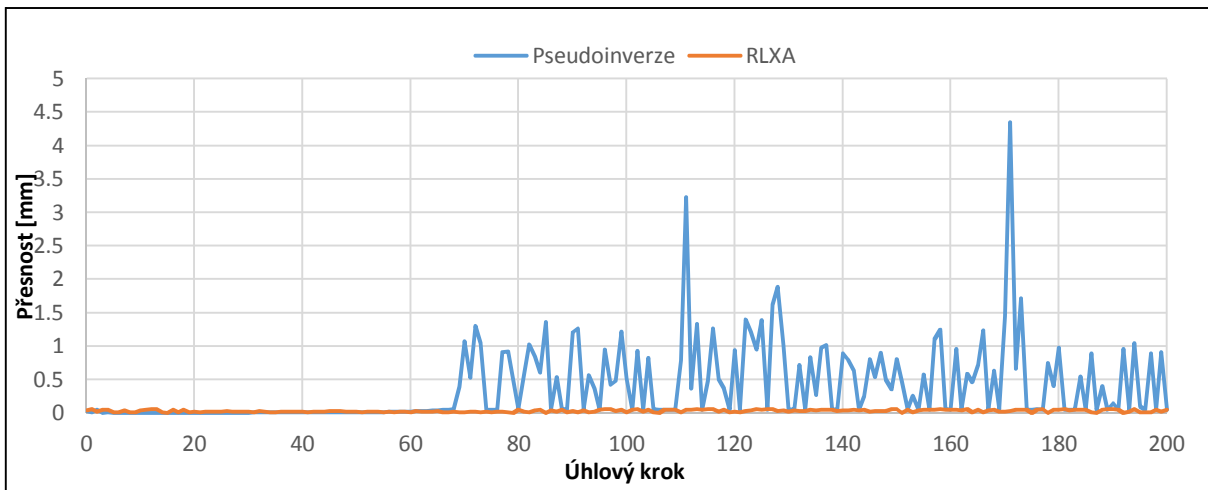


Obr. 44: Čas výpočtu jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z mimo singulární polohu, úhlový krok $\pi/25$

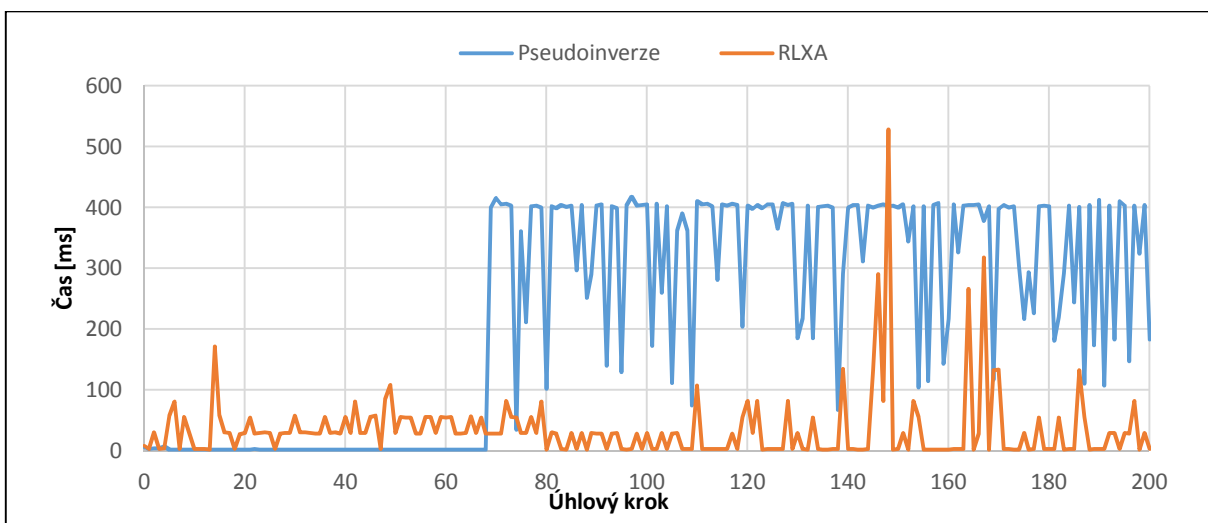
Daleko zajímavější průběhy získáme pro případ počátečního postavení robota v singulární poloze s malým úhlovým krokem výpočtu. Jak už bylo jednou řečeno, pro tuto konfiguraci narůstá čas výpočtu pomocí pseudoinverze a naopak klesá čas výpočtu dosažený metodou relaxace úhlu. Zhoršený čas výpočtu není v tomto případě jediným problémem pseudoinverze. Pseudoinverze začíná špatně sledovat předepsaný průběh žádané hodnoty (viz obr. 45). Na průběhu přesnosti jednotlivých úhlových kroků je vidět, že pro část kružnice jsou odchylky i v řádu jednotek milimetrů (viz obr. 46). Na průběhu času je vidět nárůst, který dosahuje při výpočtu jednoho úhlového kroku meze až 400 ms (viz. obr. 47). Zhruba na této hranici se výpočet přeruší kvůli překročení maximálního povoleného množství iterací. Bohužel zvyšování počtu iterací, a nastavení vyšší přesnosti, již výsledné chování výrazně nezlepší. Také není větší rozdíl mezi metodou inverze Jacobiho matice a metodou Levenberg-Marquardt. Koeficient tlumení spíše toto chování zhoršuje. Při použití metody relaxace úhlu tyto problémy nemáme. Můžeme pozorovat různé časové nároky, ale sledování žádané hodnoty funguje vždy dobře. Do této chvíle jsme se zabývali především přesností a časem výpočtu při sledování předepsané trajektorie. Neméně důležitou vlastností je také spojitost získaných úhlů natočení jednotlivých os, tzn. aby jednotlivé kroky θ_k na sebe co nejvíce navazovaly. Pokud se podíváme na požadované průběhy úhlů natočení jednotlivých os v případě, kdy má pseudoinverze problém se sledováním žádané hodnoty, tak zde tato spojitost chybí (viz obr. 48, 49). V porovnání s metodou relaxace úhlu je průběh více stochastický. Takový průběh není možné použít jako žádanou hodnotu pro regulační subsystém. Na obrázku 50, 51 je ještě uveden průběh mimo singulární polohu robota. Můžeme si všimnout, že metoda relaxace úhlu generuje spojitě průběhy v obou případech. Na obrázku 48, 49, 50 a 51 jsou uvedeny pro přehlednost pouze průběhy pro první tři osy robota, tzn. pro část kterou nazýváme manipulátor. Pro osy zápěstí je chování obdobné.



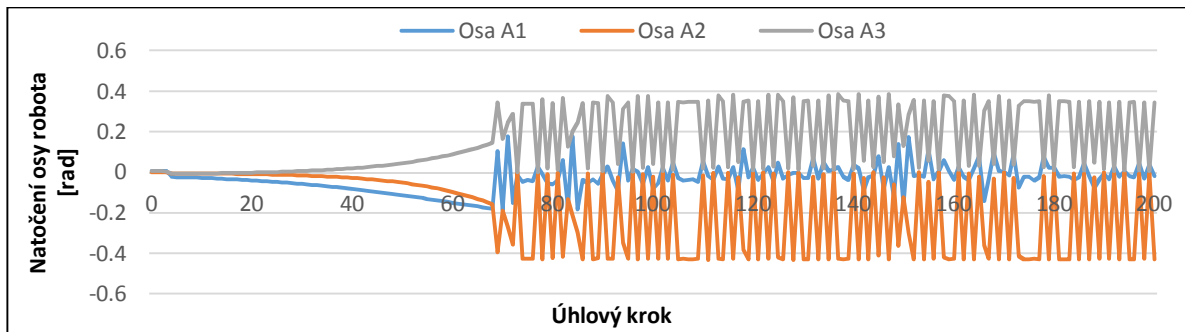
Obr. 45: Sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt



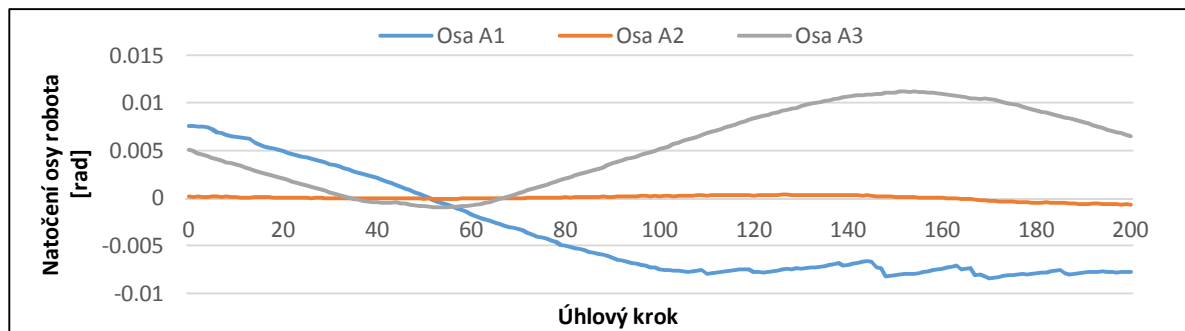
Obr. 46: Přesnost jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt



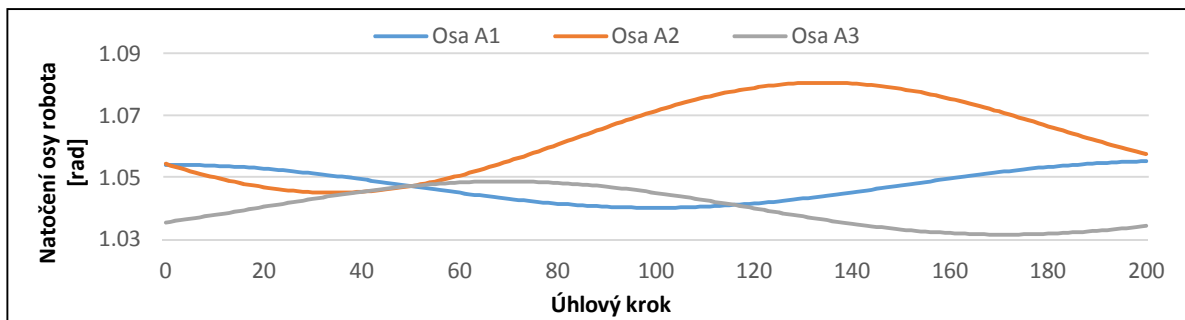
Obr. 47: Čas výpočtu jednoho úhlového kroku při sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem $\pi/100$, metoda Levenberg-Marquardt



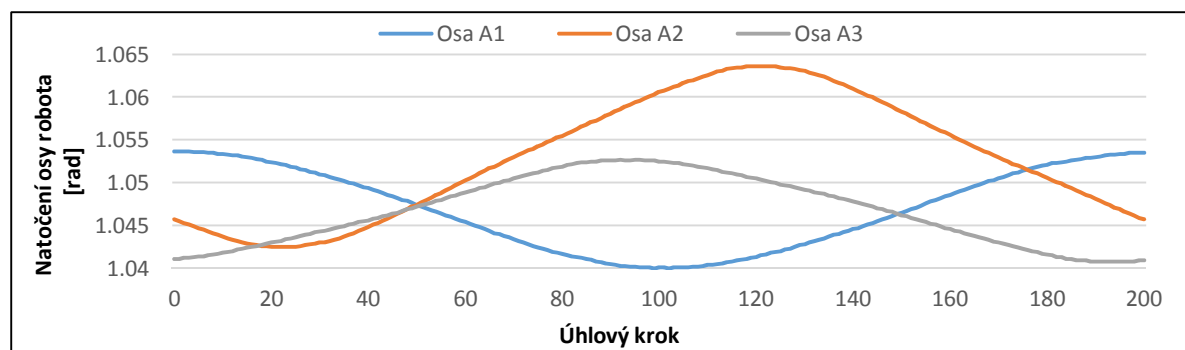
Obr. 48: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením v singulární poloze, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (Pseudoinverze)



Obr. 49: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením v singulární poloze, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (RLXA)



Obr. 50: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením mimo singulární polohu, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (Pseudoinverze)



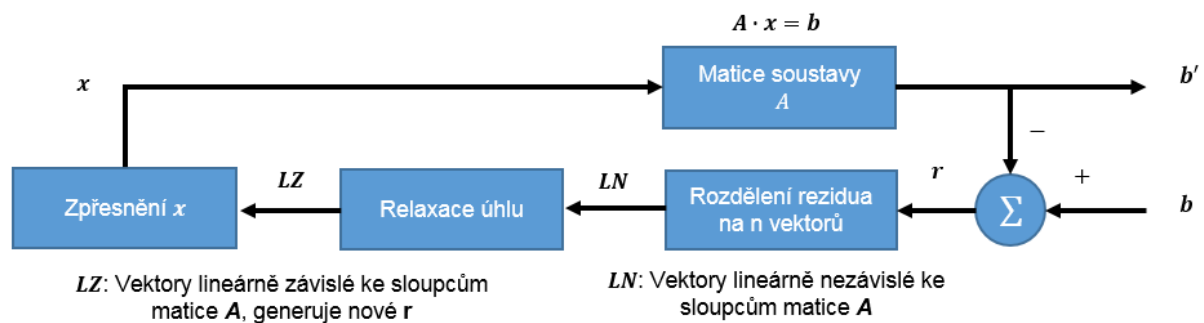
Obr. 51: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením mimo singulární polohu, poloměr kružnice 10 mm, úhlový krok $\pi/100$, metoda Levenberg-Marquardt (RLXA)

V této kapitole, věnované porovnání vybraných numerických metod pro řešení inverzní kinematické úlohy v robotice, jsme si ukázali základní vlastnosti a chování metody relaxace úhlu. Pokud nebudeme uvažovat oblast singulárního postavení kinematické struktury, je metoda relaxace úhlu plnohodnotnou náhradou standardně používané pseudoinverze. V oblasti singulárního postavení kinematické struktury můžeme pozorovat rozdílné chování obou metod. Standardně používaná pseudoinverze vykazuje lepší časové výsledky, ale zhoršené chování sledování žádané hodnoty při zmenšování diskretizačního kroku plánování trajektorie. Oproti tomu metoda relaxace úhlu je v oblasti singularity, co se týče sledování žádané hodnoty, více stabilní. Není zde ani problém s větší ztrátou spojitosti řešení. Na druhou stranu musíme očekávat určitý nárůst výpočetního času.

V tuto chvíli musíme odpovědět na otázku, co nám přináší metoda relaxace úhlu v oblasti robotiky oproti pseudoinverzi. Ačkoliv je pseudoinverze velmi dobrou obecnou numerickou metodou, z hlediska řídicího subsystému je daleko zajímavější metoda relaxace úhlu. První důvod jsme již viděli a diskutovali na obr. 48 až 51. Je to daleko lepší stabilita a spojitost při sledování žádané hodnoty trajektorie v oblasti singulární polohy kinematické struktury. Tato výhoda je zakořeněna v základním principu metody relaxace úhlu, tzn. neustále minimalizovat reziduum a po malých navazujících krocích se přibližovat k cíli. Pro oblast singularity je typické, že pro malé změny žádané hodnoty můžeme očekávat velké rozdíly $\Delta\theta$. My však počítáme s omezenou přesností, a metoda relaxace úhlu nemusí na tyto velké změny řešení ani zareagovat. Co je v obecné matematice pro numerické metody nevýhodou, zde paradoxně přispívá k potlačení nežádoucích skokových změn při sledování žádané trajektorie. To může být v robotice důležitější chování, než samotný výpočetní čas. Oproti tomu pseudoinverze nám najde přesné řešení, i když budou mezi jednotlivými kroky velké změny. To může vést v konečné instanci ke zbytečnému rozkmitání výsledné trajektorie.

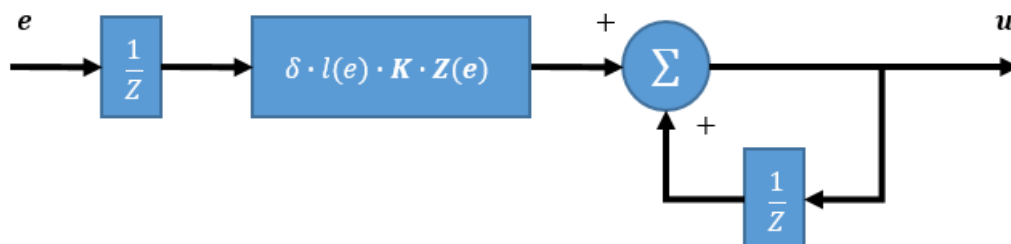
Další zajímavou vlastností metody relaxace úhlu je, že ve své podstatě odpovídá zpětnovazebnímu zapojení regulátoru. Tato vlastnost ještě více podporuje myšlenku využití v regulačním subsystému. Na obrázku 52 je znovu znázorněna základní myšlenka metody relaxace úhlu ve zpětnovazebním zapojení. Pokud se ještě podíváme na zápis algoritmu v maticové podobě $\mathbf{P}_{k+1} = \sigma_k \cdot l_k \cdot \mathbf{K} \cdot \mathbf{Z}_k + \mathbf{P}_k$, tak bychom mohli najít podobnost v PSD regulátoru v přírůstkovému tvaru, resp. jedné jeho části. Rozdíl je ovšem v tom, že metoda relaxace úhlu pracuje s MIMO (Multiple Input Multiple Output) systémem a PSD je v základním tvaru určen pro SISO (Single Input Single Output) systém. Přírůstkový tvar PSD regulátor má podobu $\mathbf{u}_{k+1} = \mathbf{u}_k + q_0 \cdot e_{k+1} + q_1 \cdot e_k + q_2 \cdot e_{k-1}$. Pozn.: pro názornost jsme označili část podobající se metodě relaxace úhlu tučně, tzn. nejedná se v tomto případě o vektory. Potom \mathbf{P} odpovídá pomyslně u a $l \cdot \mathbf{Z}$ zase e . \mathbf{Z} matice \mathbf{P} vyjadřujeme \mathbf{x} , což odpovídá akční veličině (viz obr. 53). V proměnných $l \cdot \mathbf{Z}$ se zase skrývá reziduum, tzn. regulační odchylka. Matice \mathbf{K} reprezentuje model systému, resp. jeho matici zesílení. A náhodné σ zde figuruje jako určitá konstanta proporcionality. Metoda relaxace úhlu je vlastně speciální regulátor pro MIMO systémy s okamžitou odezvou, které jsou definovány pouze maticí zesílení, resp. maticí soustavy.

Nebudeme zde podrobně rozebírat chování metody relaxace úhlu jako regulátoru nějakého obecného systému. Na základě několika mála testů můžeme říci, že se jedná o regulátor poměrně konzervativní, tzn. dosažení žádané hodnoty je pozvolné. V proporcionalitě jsme totiž omezeni intervalem (0,1) náhodné proměnné δ . Hodnoty mimo tento interval regulační pochod velmi lehce rozkmitají. Pokud přidáme do systému ještě dynamiku, tak pro rychlé systémy je metoda relaxace úhlu stále funkční. Se zvyšujícím se zpožděním systému dochází k postupnému rozkmitání regulačního pochodu, až dojde k jeho nestabilitě. Dynamika se dá ovšem také zapracovat do metody relaxace úhlu, a to přímo do matice K , která odpovídá modelu systému. Tím by šlo regulátor více zobecnit pro systémy s různou dynamikou. To je ovšem otázka dalších testů a výzkumu.



Obr. 52: Zpětnovazební zapojení metody relaxace úhlu

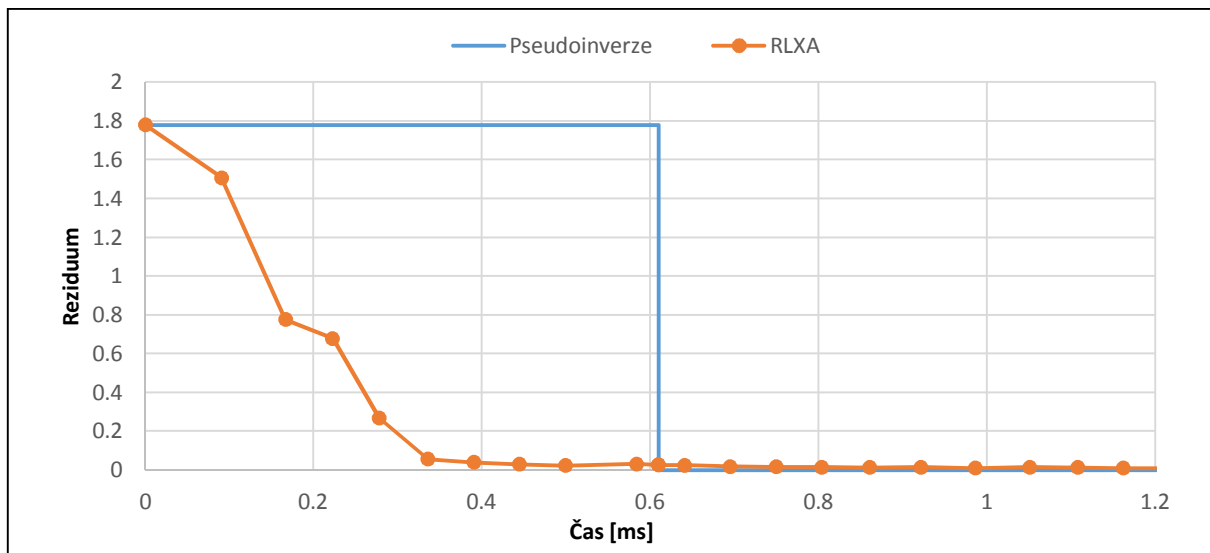
Pokud ještě zůstaneme u metody relaxace úhlu jako regulátoru, můžeme jej v principu vyjádřit pomocí z-transformace (grafické vyjádření viz obr. 53, pozn.: signály mezi bloky jsou vektory). Potom jsme schopni tento diskretní model převést pomocí bilineární Tustinovy transformace do spojitě podoby. Teoreticky jsme schopni řešit výpočet soustavy lineárních rovnic s vlastnostmi pseudoinverze i pro spojitě signály. Teoreticky by pak bylo možné sestavit elektrický obvod, který by celou úlohu dovedl řešit. Toto je téma, které by bylo nutné objasnit dalším výzkumem, ale v principu to jistě možné je.



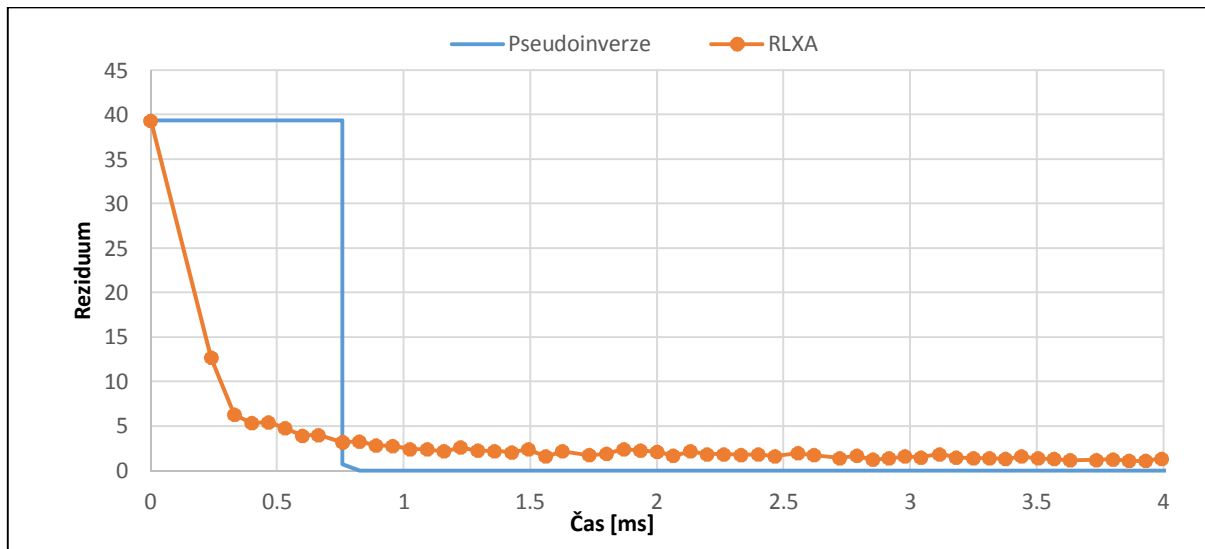
Obr. 53: Metoda relaxace úhlu vyjádřená pomocí z-transformace

Posledním důležitým faktorem, který zdůvodňuje smysl použití metody relaxace úhlu, je možnost získání velmi rychlého odhadu řešení. V rámci časového porovnání obou metod jsme v experimentech požadovali dosažení předepsané přesnosti. Nebudeme teď brát na zřetel nějakou konkrétní hodnotu přesnosti, ale podíváme se na obě metody z hlediska jejich průběhu. Z pseudoinverze

nezískáme výsledek hned. Musíme počkat, až bude výpočet hotový. Z metody relaxace úhlu můžeme získat odhad řešení v každém kroku iterace. Tento odhad se samozřejmě s počtem kroků zpřesňuje. Nejlépe to vystihuje obrázek 54 a 55, kde je znázorněn časový průběh rezidua $\max|\mathbf{r}_k|$ pro obě metody. Na obr. 54 je vzorový průběh pro matici $n = 2$ a na obr. 55 pro matici $n = 10$. Rozměry matice se opět pohybují v námi stanovené pomyslné mezi fyzikální realizovatelnosti. Z grafů můžeme vidět, jak metoda relaxace úhlu zprvu velmi prudce minimalizuje reziduum, dokud nedojde ke zlomu a zpomalení. Následně dochází k pozvolnému přibližování k nule. U pseudoinverze se až po získání přesného výsledku nic neděje. Můžeme tedy využít první fáze metody relaxace úhlu k rychlému odhadu řešení a získat tak nějaké první hodnoty dříve, než skončí samotný výpočet pseudoinverze. Není ani vyloučeno obě metody kombinovat. Pokud bychom naše časové porovnání provedli pro nižší přesnosti, začne být metoda relaxace úhlu rychlejší než pseudoinverze. Samozřejmě pokud dodržíme určité předpoklady, především omezení velikosti matice $n \leq 10$ a výpočet mimo oblast singulární polohy kinematické struktury.



Obr. 54: Časový průběh minimalizace rezidua metodou relaxace úhlu a pseudoinverze, matice $n = 2$



Obr. 55: Časový průběh minimalizace rezidua metodou relaxace úhlu a pseudoinverze, matice $n = 10$

Tato disertační práce se jmenuje rychlé heuristické metody numerického řešení inverzní kinematiky. Na základě předchozí diskuze je již zřejmé, proč metodu relaxace úhlu můžeme považovat také v určitých případech za rychlou. Je to právě fakt, že jsme schopni velmi rychle dostat poměrně dobrý odhad řešení, a to při zachování výpočetních vlastností, na které jsme zvyklí u pseudoinverze. Na základě našeho porovnání jsme zjistili následující chování obou metod:

- metodu relaxace úhlu lze použít pro numerické výpočty inverzní kinematické úlohy v robotice v kombinaci s metodou inverze Jacobiho matice a metodou Levenberg-Marquardt. Metodu relaxace úhlu nelze kvůli špatným výsledkům kombinovat s Newtonovou metodou,
- mimo oblast singulárního postavení kinematické struktury jsou časové výsledky metody relaxace úhlu téměř srovnatelné s výsledky pseudoinverze,
- pro oblast singulárního postavení kinematické struktury a větší diskretizační krok jsou časové výsledky metody relaxace úhlu horší, než výsledky pseudoinverze,
- pro oblast singulárního postavení kinematické struktury se při snižování diskretizačního kroku časové výsledky metody relaxace úhlu zlepšují a u pseudoinverze zhoršují,
- pro oblast singulárního postavení kinematické struktury dochází při snižování diskretizačního kroku u pseudoinverze ke ztrátě spojitosti a špatnému sledování trajektorie. Metoda relaxace úhlu je proti tomuto chování více odolná,
- při snižování požadované přesnosti může být metoda relaxace úhlu rychlejší, než pseudoinverze,
- metoda relaxace úhlu může být použita pro rychlý odhad řešení,
- metoda relaxace úhlu odpovídá struktuře zpětnovazebního regulátoru.

3.6 Praktická aplikace

Kromě teoretických experimentů v prostředí MATLAB si nyní ukážeme použití metody relaxace úhlu v praktické aplikaci v rámci softwaru SKODA TOOL. Bude se jednat o časovou optimalizaci robotové trajektorie. Často se stane, že programátoři nemají zkušenosti s tvorbou trajektorie a chováním reálného robota. Hotové programy tak nemusí být časově optimální. Z tohoto důvodu se prostřednictvím softwaru SKODA TOOL snažíme hledat časové potenciály na již zprovozněném zařízení. Časová ztráta může být způsobena různými věcmi, např. použitím zbytečně dlouhého zpoždění v logice programu, nevhodným otevřením kleští při svařování, nebo špatnými parametry dráhy. K nalezení těchto časových potenciálů nepotřebujeme žádné složité výpočty. V principu stačí hledat nestandardně vysoké nebo nízké hodnoty vybraných parametrů. Daleko náročnější úkol je úprava robotové trajektorie. Ukážeme si nyní, podle jakého principu hledáme lepší časové průběhy trajektorie v softwaru SKODA TOOL.

Trajektorie robota KUKA je popsána pomocí bodů v kartézském souřadnicovém systému, tzn. pomocí šesti parametrů \mathbf{T}_R . Pokud použijeme převod do podoby homogenní matice transformace, tak pomocí matice \mathbf{T}_H . Všechny hodnoty jsou uvedeny v záloze robota v textové podobě v souborech s příponou .dat. Dále je zde informace o použitém nástroji \mathbf{T}_T a použité bázi \mathbf{T}_B . Pozn.: \mathbf{T}_T a \mathbf{T}_B uvažujeme již v podobě homogenní matice transformace. Potom přímá kinematická úloha v robotice je dána vztahem 67. Denavit-Hartenbergovy parametry můžeme vyčíst opět ze zálohy robota ze strojních dat uložených v souboru machine.dat. Strojní data odpovídají použitému modelu robota, takže získáme vždy odpovídající transformaci \mathbf{T}_H .

$$\mathbf{T}_H = \mathbf{T}_B^{-1} \cdot \mathbf{T}_1 \cdot \mathbf{T}_2 \cdot \mathbf{T}_3 \cdot \mathbf{T}_4 \cdot \mathbf{T}_5 \cdot \mathbf{T}_6 \cdot \mathbf{T}_T \quad (67)$$

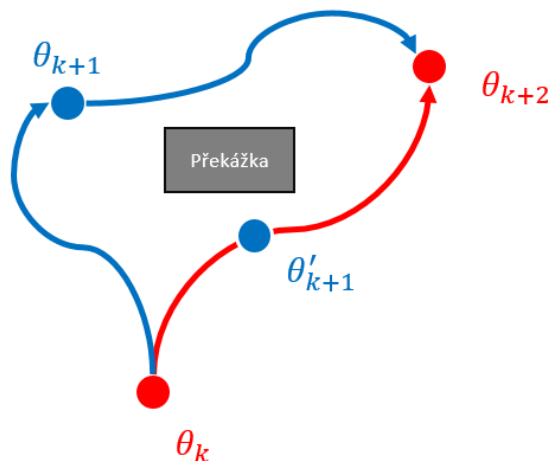
Matice \mathbf{T}_T a \mathbf{T}_B jsou známé, tudíž můžeme vztah 67 zjednodušit vynásobením jejich inverzí $\mathbf{T}_H = \mathbf{T}_B \cdot \mathbf{T}_H \cdot \mathbf{T}_T^{-1} = \mathbf{T}_1 \cdot \mathbf{T}_2 \cdot \mathbf{T}_3 \cdot \mathbf{T}_4 \cdot \mathbf{T}_5 \cdot \mathbf{T}_6$. Najít inverzní matici \mathbf{T}_T^{-1} není problém. Obecně pro homogenní matici transformace platí, že matice rotace je ortogonální antisymetrická. Potom $\mathbf{R}^{-1} = \mathbf{R}^T$. Pro inverzní matici homogenní transformace pak platí vztah 68.

$$\mathbf{T}_T^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \cdot \mathbf{P} \\ 0 & 1 \end{bmatrix} \quad (68)$$

V tuto chvíli jsme se dostali do stavu, který je nám znám z experimentů s robotem KUKA KR210 R2700 EXTRA. Pomocí metody relaxace úhly vypočítáme inverzní kinematickou úlohu v robotice a stanovíme natočení jednotlivých os θ .

Převod do prostoru kloubových souřadnic nám umožní zjistit, které body skýtají potenciál pro časovou optimalizaci. Při hledání nás bude zajímat pohyb robota osově specifickým způsobem. Lineární

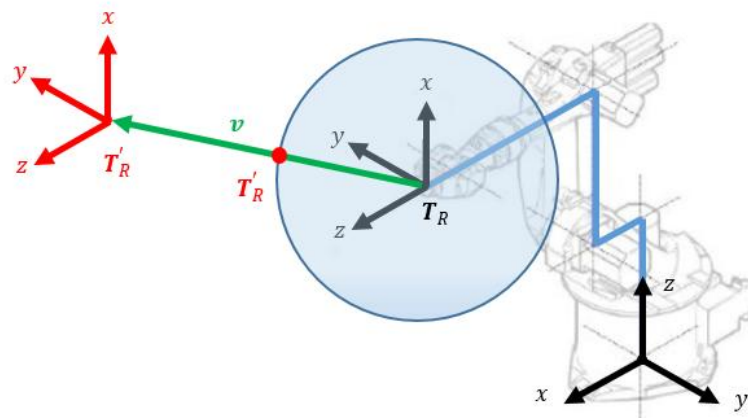
a cirkulární pohyb nás nebude zajímat, jelikož se jedná o synchronizovaný pohyb všech os po předepsané dráze. Zde můžeme upravit maximálně rychlost a akceleraci. Zatímco osově specifický pohyb je nesynchronizovaný a také nejrychlejší pohyb do dalšího předepsaného bodu. To znamená, že řídicí systém robota dostane informaci o nové pozici θ_{k+1} , a pak už záleží na vlastnostech každé osy, jak rychle dosáhne žádané hodnoty. V praxi to ovšem takto nebývá. Aby se šetřila mechanika robota, zpomalí se jednotlivé osy podle té nejpomalejší tak, aby všechny dosáhly žádané hodnoty ve stejný čas. Tím, že se osy nepřestavují maximální rychlostí, se robot zbytečně nenamáhá. Na základě katalogové rychlosti jednotlivých os robota ω_i pak můžeme určit časový průběh z θ_k do θ_{k+1} . Použijeme závislost dráhy, času a rychlosti $\Delta\theta_i = \omega_i \cdot t_i$. Z rovnice vypočteme t_i a určíme, která osa bude nejpomalejší. Podle nejpomalejší osy pak zkorigujeme rychlosti zbývajících os, aby žádaná pozice byla dosažena ve stejný čas. Nyní známe nejrychlejší průběh mezi dvěma libovolnými body v čase t . Je nutné poznamenat, že zanedbáváme akcelerační a decelerační rampu. Ta se může do výpočtu také zahrnout.



Obr. 56: Průběh ideální (červená) a předepsané (modrá) trajektorie

Trajektorie robota není nikdy popsána pouze dvěma body. Mezi počátečním a koncovým bodem jsou většinou další pomocné body, které dokreslují celou trajektorii. Velmi často se musíme při pohybu robota vyhnout např. nějaké překážce viz obr. 56. Nyní budeme chtít zjistit, jestli pomocný bod θ_{k+1} vnáší do trajektorie nějaké časové zpoždění. Podle výše popsaného výpočtu stanovíme nejrychlejší možný čas z θ_k do θ_{k+2} (ideální červená trajektorie). Potom stanovíme čas z θ_k do θ_{k+1} a z θ_{k+1} do θ_{k+2} (předepsaná modrá trajektorie). Pokud bude čas modré trajektorie větší, než čas ideální červené trajektorie, můžeme se pokusit o časovou optimalizaci bodu θ_{k+1} . Tímto způsobem pracuje rutina pro odhalování časových potenciálů v softwaru SKODA TOOL. Obecně neplatí, že přidavný bod vnáší do trajektorie nějaké zpoždění. Stává se, že pro dlouhé přejezdy robota zůstává čas nejpomalejší osy nezměněn. Ideální jsou pohyby robota na malém prostoru, kde jsou přestavovací časy jednotlivých os hodně podobné.

Pokud zjistíme, že bod θ_{k+1} je možné optimalizovat, pokusíme se nalézt nejrychlejší ideální postavení θ'_{k+1} . Jedná se o bod, který dosáhneme ve stejném čase jako θ_{k+1} , ovšem na červené ideální trajektorii. Pokud neexistuje mezi bodem θ_{k+1} a θ'_{k+1} žádná překážka, můžeme bez problémů θ'_{k+1} prohlásit za nové θ_{k+1} . Získáme tím nejrychlejší možnou trajektorii z bodu θ_k do θ_{k+2} . V případě, že se mezi bodem θ_{k+1} a θ'_{k+1} vyskytuje nějaká překážka, musíme provést pouze částečnou transformaci. Přiblížíme θ_{k+1} ve směru bodu θ'_{k+1} tak, aby zároveň nedošlo ke kolizi s překážkou. V prostoru kloubových souřadnic se transformace špatně omezuje. Daleko lepší je omezit polohu koncového bodu robota v kartézském souřadnicovém systému. Převědeme proto θ_{k+1} a θ'_{k+1} pomocí přímé kinematické úlohy v robotice zpět do podoby homogenní matice transformace T_H a T'_H , resp. do podoby T_R a T'_R (viz obr. 57). V této podobě můžeme omezit translaci a rotaci koncového bodu robota. Samotná transformace je velice jednoduchá. Určíme vektor $v = T'_R - T_R$, na který aplikujeme omezující podmínky. Tím určíme nový omezený vektor v_o . Potom $T'_R = T_R + v_o$. K nalezení bodu T'_R můžeme použít i metodu relaxace úhlu. Úlohu můžeme popsat soustavou lineárních rovnic, kterou následně řešíme. Až se algoritmus dostane na předepsanou mez, iteraci ukončíme. Omezením intervalu náhodného δ můžeme zmenšit iterační krok a zamezit tak překročení omezující podmínky v rámci jedné iterace.



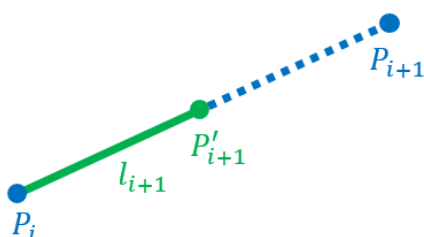
Obr. 57: Přesun bodu T_R k ideální trajektorii s omezením

Volba omezujících podmínek není v softwaru SKODA TOOL automatizovaná a provádí se ručně. Je nutné zvážit aktuální postavení robota a umístění překážek v okolí. Dále je nutné upravenou trajektorii zkontrolovat. Ujistit se o úspoře času a bezkolizním průběhu. Z tohoto důvodu se roboty takto plošně neoptimalizují. Optimalizují se pouze roboty, u kterých vznikne potřeba a jiné časové potenciály byly již vyčerpány.

4 Metoda relaxace délky

4.1 Popis metody relaxace délky

Druhým diskutovaným algoritmem disertační práce bude heuristická metoda relaxace délky, která je přímo určena pro řešení inverzní kinematické úlohy v robotice. Základním kamenem algoritmu bude operace, kterou nazveme relaxace délky. Relaxace délky vznikla na základě úpravy metody relaxace úhlu. Opět si na příkladu vysvětlíme, co tato operace znamená. Uvažujme dva libovolné body P_i a P_{i+1} (viz obr. 58). Dále máme definovanou délku l_{i+1} . Celý princip relaxace délky spočívá v přesunutí bodu P_{i+1} ve směru $\mathbf{v}_{i+1} = P_{i+1} - P_i$ do vzdálenosti l_{i+1} od bodu P_i tak, že nám vznikne nový bod P'_{i+1} .



Obr. 58: Relaxace délky

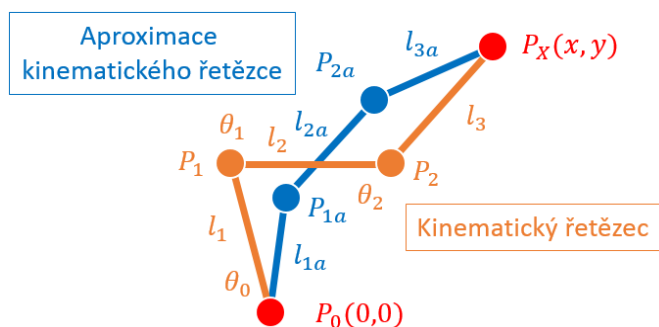
Definice 4.1. Relaxaci délky definujeme jako přesunutí bodu P_{i+1} ve směru $\mathbf{v}_{i+1} = P_{i+1} - P_i$ tak, aby metrika $\rho_e(P_i, P'_{i+1}) = l_{i+1}$ (69). Tento jednoduchý výpočet bude základním stavebním kamenem celého algoritmu.

$$P'_{i+1} = \frac{\mathbf{v}_{i+1}}{\|\mathbf{v}_{i+1}\|_e} l_{i+1} + P_i \quad (69)$$

Můžeme si všimnout, že vztah 68 je velmi podobný vztahu 5 pro relaxaci úhlu. Obě metody používají podobné matematické operace a jsou si v tomto ohledu velice blízké. Zajímavé také je, že obě metody můžeme použít k řešení inverzní kinematické úlohy v robotice, jen každý vztah použijeme jiným způsobem.

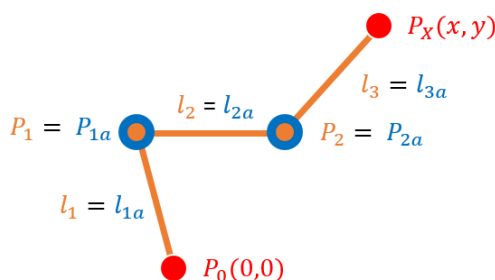
Nyní si vysvětlíme podstatu metody relaxace délky. Uvažujme následující situaci. Máme kinematický řetězec s třemi rotačními vazbami v bodech (P_0, P_1, P_2) a tři kinematické spoje o délce (l_1, l_2, l_3) . Tato situace je znázorněna oranžovou barvou na obr. 59. Pozice počátečního bodu P_0 a pozice koncového bodu P_x je známa. Známé body jsou označeny červenou barvou. Pozice bodů P_1 a P_2 není známa. Naším úkolem je určit úhel rotačních vazeb $(\theta_0, \theta_1, \theta_2)$ v bodech (P_0, P_1, P_2) .

V prvním kroku si určíme aproximační kinematický řetězec s náhodně umístěnými body P_{1a} a P_{2a} . Délka kinematických spojů není stejná $(l_1 \neq l_{1a}, l_2 \neq l_{2a}, l_3 \neq l_{3a})$. Aproximační kinematický řetězec je znázorněn modrou barvou na obr. 59.



Obr. 59: Aproximační kinematický řetězec

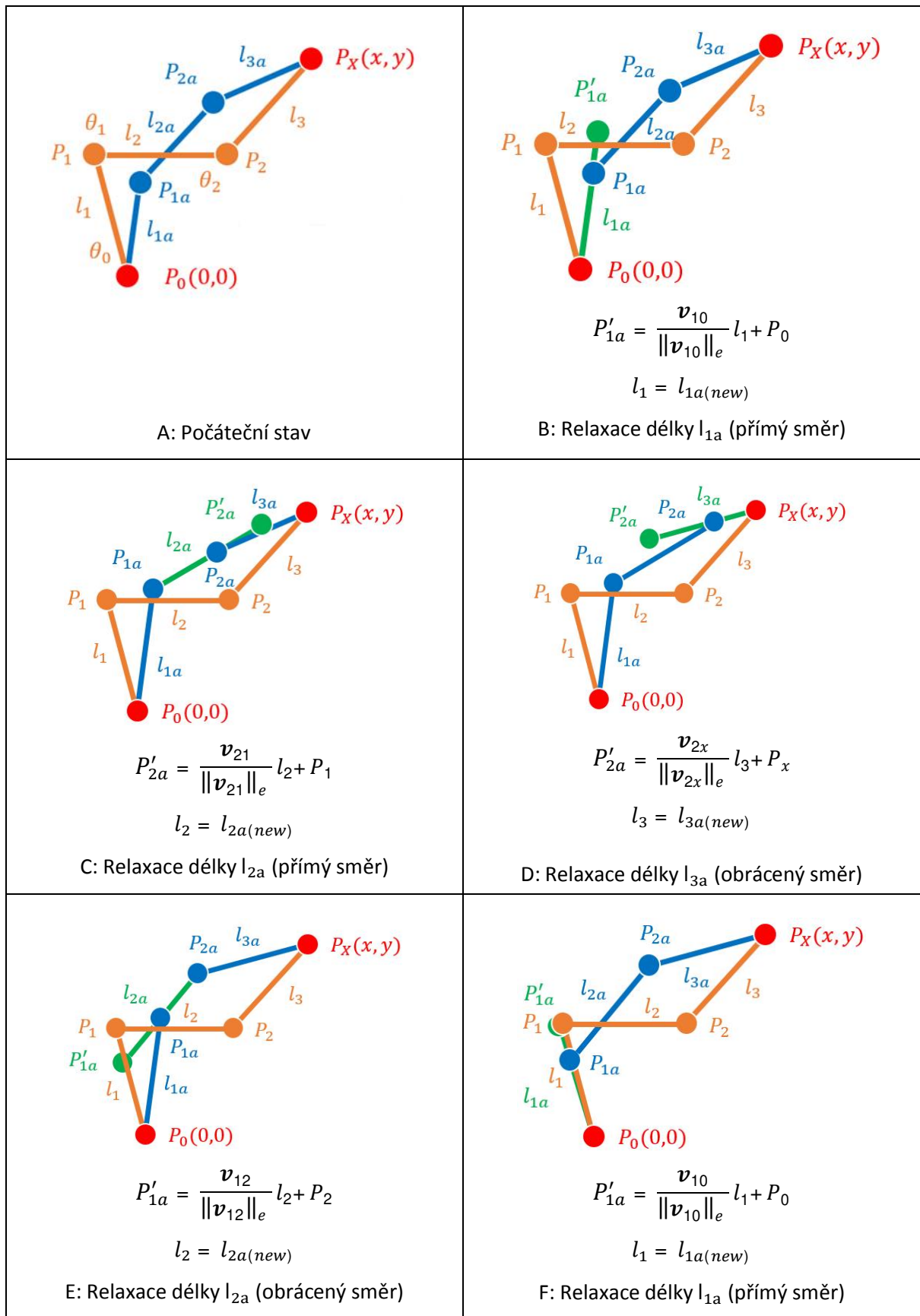
Hlavní myšlenka metody relaxace délky je následující. Pokud nalezneme takové body P_{1a} a P_{2a} aproximačního kinematického řetězce, že $l_1 = l_{1a}$, $l_2 = l_{2a}$ a $l_3 = l_{3a}$, pak tyto body můžeme prohlásit za P_1 a P_2 . Celková délka obou řetězců je pak stejná a počáteční bod P_0 a koncový bod P_x je spojen stejnou kinematickou strukturou. Následně můžeme určit úhly všech rotačních kloubů (θ_0 , θ_1 , θ_2), jelikož známe pozici všech bodů aproximačního kinematického řetězce ($P_0, P_1 = P_{1a}, P_2 = P_{2a}, P_x$). Optimální řešení je znázorněné na obr. 60.



Obr. 60: Optimální řešení metody relaxace délky

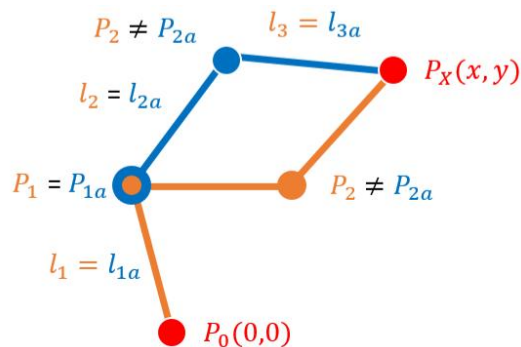
Je zde důležitá otázka, jak upravit náhodnou počáteční pozici bodů P_{1a} a P_{2a} aproximačního kinematického řetězce a dosáhnout tak podmínky rovnosti délek $l_1 = l_{1a}$, $l_2 = l_{2a}$ and $l_3 = l_{3a}$. K tomuto účelu nám právě poslouží relaxace délky.

Algoritmus relaxace délky funguje tak, že postupně procházíme aproximační kinematický řetězec a relaxujeme všechny délky kinematických spojů. To znamená, že pokud délka l_i kinematického řetězce není rovna odpovídající délce l_{ia} aproximačního kinematického řetězce $l_i \neq l_{ia}$, použijeme na bod P_{ia} relaxaci délky z definice 4.1. Body, u kterých známe jejich pozici (v našem případě P_0 a P_x), nepřesouváme. Pokud narazíme při průchodu aproximačním kinematickým řetězcem na bod se známou pozicí, začneme procházet aproximační kinematický řetězec od tohoto bodu obráceně (přímý a obrácený směr). Prvních několik kroků algoritmu je ukázáno na situaci kinematického řetězce z obrázku 59 (viz obr. 61).



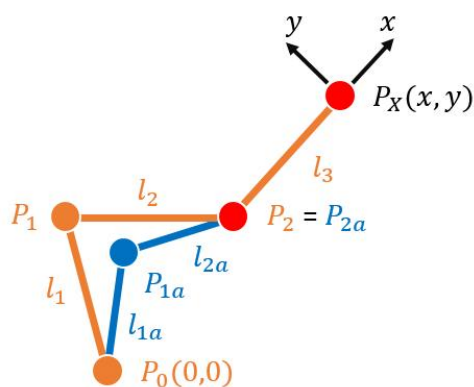
Obr. 61: Postupná relaxace aproximačního kinematického řetězce

Po několika průchodech aproximačním kinematickým řetězcem dosáhneme finální pozice bodů P_{1a} a P_{2a} (viz obr. 62). Celková délka obou kinematických řetězců je sice stejná, ale orientace rotačních vazeb se v tomto případě nerovná. Tento výsledek můžeme nazvat jako neoptimální.



Obr. 62: Výsledek metody relaxace délky (neoptimální)

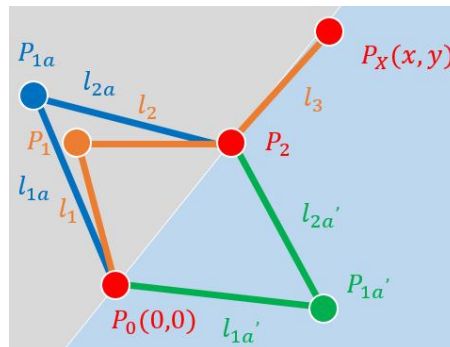
Můžeme se pokusit dosáhnout optimálního řešení a zachovat orientaci všech rotačních vazeb. První problém, který můžeme jednoznačně vyřešit, pokud je koncový bod P_x definován včetně své rotace, je pozice předposledního bodu P_2 . Stačí se ve správném směru vrátit o vzdálenost l_3 zpět, jak je vidět na obrázku 63. Pokud vyjádříme přímou kinematickou úlohu v robotice, pak můžeme z výsledné homogenní matice transformace T_H převzít odpovídající směr. V tomto případě se jedná o směr osy x , což odpovídá prvnímu sloupci T_H .



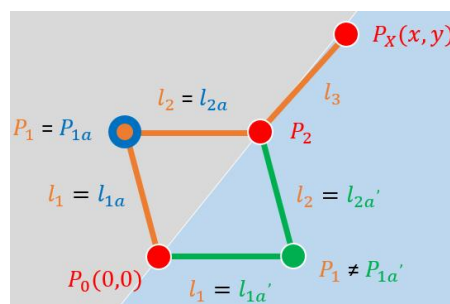
Obr. 63: Výpočet pozice předposledního bodu P_2

Pokud nyní použijeme metodu relaxace délky na situaci z obrázku 63, orientace posledního rotačního kloubu bude zachována. Na druhou stranu, stále zde zůstávají dvě možná řešení, která závisí na volbě počáteční pozice aproximačního řetězce v bodu P_{1a} . Volbou počáteční pozice můžeme ovlivnit výsledné řešení inverzní kinematické úlohy v robotice. Metoda relaxace délky má snahu konvergovat k nejbližšímu možnému řešení. Vše záleží na tom, jakou rovinu (šedou, nebo modrou viz. obr 64) pro volbu počátečního bodu zvolíme. Aproximační kinematický řetězec s počátečním bodem P_{1a} (modrá barva) konverguje přímo do bodu P_1 (optimální řešení). Aproximační kinematický řetězec s počátečním bodem P_{1a}' (zelená barva) konverguje do druhého možného řešení, které se ovšem nerovná bodu P_1 (neoptimální řešení). Obě možná řešení jsou znázorněna na obrázku 65. Ještě je nutné poznamenat, že

pokud by byl počáteční bod aproximace posazen na rozhraní obou rovin, tak metoda relaxace délky nebude konvergovat. Bod P_{1a} pak bude oscilovat mezi bodem P_0 a P_2 .

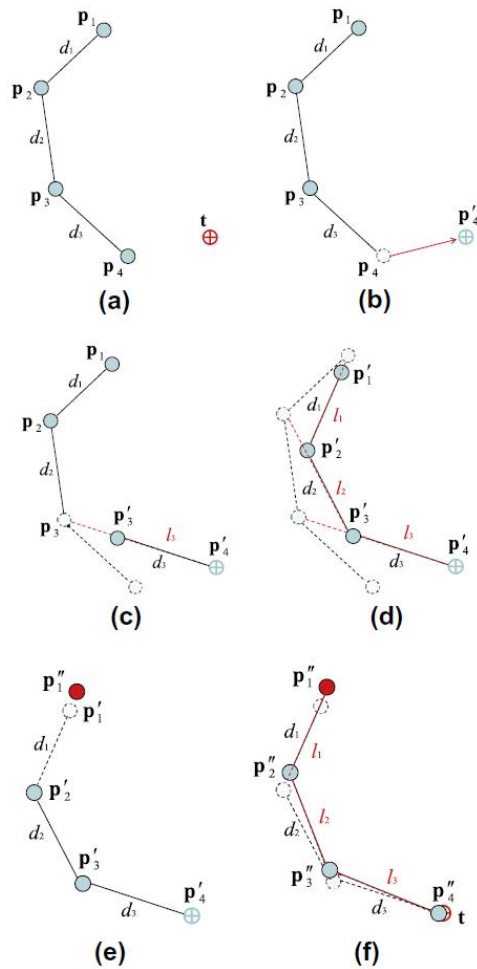


Obr. 64: Volba počátečních podmínek



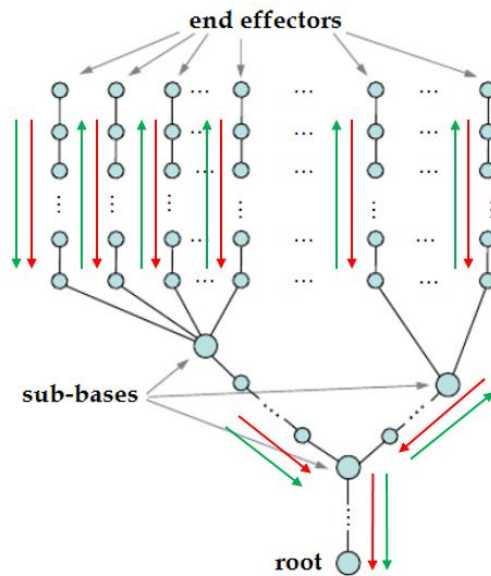
Obr. 65: Řešení inverzní kinematické úlohy dle počátečních podmínek

Metoda relaxace délky vznikla nezávisle jako variace metody relaxace úhlu. Pokud prostudujeme známé heuristické metody pro řešení inverzní kinematické úlohy v robotice, zjistíme, že existuje metoda postavená na stejném principu, která nese název FABRIK (Forward And Backward Reaching Inverse Kinematics). FABRIK publikoval v roce 2011 Andreas Aristida a Joan Lasenby z univerzity v Cambridge [29]. Tím, že byla metoda relaxace délky testována nezávisle, jsou zde ovšem principiální rozdíly. Metodu relaxace délky nebudeme za těchto okolností nutně nazývat metodou novou, ale přinejmenším vylepšením stávající heuristické metody FABRIK. Na obr. 66 je znázorněn princip hledání inverzní kinematické úlohy v robotice pomocí metody FABRIK. Základní rozdíl mezi metodou relaxace délky a FABRIK je v tom, že FABRIK pracuje i s koncovými body kinematického řetězce, zatímco u metody relaxace délky jsou tyto body fixní. Pokud použijeme terminologii z této práce, tak FABRIK relaxuje i koncové body kinematického řetězce. To je ovšem zbytečná operace, protože FABRIK následně předepisuje přesunout tento bod zpět do počátečního stavu (viz přechod mezi situací D-E na obrázku 66). Metoda relaxace délky nepřesouvá koncové body, čímž šetří výpočetní čas. Proto můžeme hovořit o vylepšené metodě FABRIK.



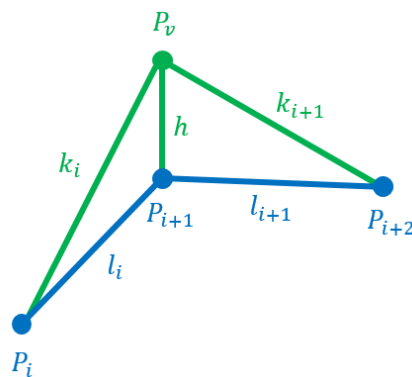
Obr. 66: Výpočet inverzní kinematické úlohy v robotice pomocí metody FABRIK, obrázek převzat z [29]

Jak metodu relaxace délky, tak metodu FABRIK můžeme použít pro výpočet paralelní kinematické struktury. Zde najdeme druhý principiální rozdíl. Obecně je nutné u obou metod nějakým způsobem procházet kinematickou strukturu tam a zpět. Metoda FABRIK v tomto ohledu rozeznává počátek kinematické struktury (root), koncové body kinematické struktury (end effectors) a vnitřní část kinematické struktury (sub bases), jak je znázorněno na obr. 67. Dle FABRIK se prochází kinematický řetězec v pořadí end effector, sub base, root a zpět. Tzn. průchody jsou rozděleny do samostatných částí, které na sebe navzájem navazují. V rámci metody relaxace úhlu nedělíme kinematickou strukturu na samostatné části, ale rozeznáváme pouze fixní (koncové a počáteční) a pohyblivé body. Potom postupujeme tak, že zvolíme první fixní bod a od něj začneme procházet kinematickou strukturu ve směru procházení grafu do šířky. Pokud narazíme na jiný fixní bod, tak procházení dané větve ukončíme. Pokud je procházení u konce, přesuneme se na druhý fixní bod a procházení zopakujeme. Z hlediska programování pak stačí vytvořit jednu obecnou rutinu pro procházení grafu do šířky, kterou pak můžeme použít na libovolnou kinematickou strukturu. Na obrázku 67 je červenou barvou označen průchod kinematickou strukturu pomocí metody FABRIK a zelenou barvou průchod pomocí metody relaxace délky. Můžeme vidět, jak se obě cesty od sebe liší.



Obr. 67: Průchod kinematickým řetězcem pomocí metody FABRIK a metody relaxace délky, obrázek převzat z [29]

Třetí, a poslední, rozdíl je v technice omezení pohybu kinematické vazby. Problém nastane, pokud začneme aplikovat obě metody na kinematické struktury, kterou jsou definované včetně třetího rozměru. Potom oba algoritmy přesouvají body, jako by se jednalo o sférickou vazbu. Pokud chceme pracovat stále s rotační vazbou, musíme omezit pohyb této vazby do vybrané roviny. Metoda FABRIK postupuje tak, že výsledek každé úpravy zobrazí do požadované roviny. Případně se do výpočtu vnáší určitá matematická omezení. Toto téma je v rámci metody FABRIK celkem rozsáhlé a více je diskutováno zde [29]. My si ukážeme náš přístup, který nevnáší do výpočtu žádný další matematický aparát. Jedná se o omezení pohybu kinematické vazby pomocnými virtuálními spoji (viz obr. 68). Tento spoj tvoří jeden virtuální bod P_v , který je v libovolné vzdálenosti h ve směru pomyslné osy rotace. Na základě Pythagorovy věty se určí velikosti ramen k_i a k_{i+1} . Takto vzniklou paralelní kinematickou strukturu budeme řešit standardní cestou pomocí metody relaxace délky. Přídavná ramena nedovolí při výpočtu, aby bod P_{i+1} ležel v jiné rovině, než body P_i a P_{i+2} . Tímto způsobem transformujeme sférickou vazbu na rotační bez nutnosti úpravy algoritmu.



Obr. 68: Transformace sférické vazby na rotační pomocí paralelního virtuálního spoje

4.2 Porovnání vybraných heuristických metod

Metodu relaxace délky ještě porovnáme s jinými heuristickými algoritmy řešení inverzní kinematické úlohy v robotice. Bude nás opět zajímat čas potřebný pro vyřešení dané úlohy. Metodu relaxace délky porovnáme s již diskutovanou metodou FABRIK a známým heuristickým algoritmem CCD. Pro tento účel byl vytvořen evaluační software v programovacím jazyce C#. Pomocí evaluačního softwaru byl definován kinematický řetězec v podobě planárního manipulátoru s libovolným počtem rotačních vazeb. Pro daný počet kinematických vazeb byl měřen čas nutný k dosažení řešení s definovanou přesností $\|\mathbf{r}\|_e < 0.01$ mm. Počet kinematických vazeb se opět pohyboval v rozmezí $2 \leq n \leq 10$. Výsledné časy jsou uvedeny v tabulce 8. Metoda relaxace délky je srovnatelná s metodou FABRIK. Pro menší počet kinematických spojů je dokonce rychlejší. To je dáno tím, že se u metody FABRIK musí relaxovat i koncové body. Tato nevýhoda se s vyšším počtem kinematických vazeb ztrácí, jelikož poměr koncových bodů vůči relaxovaným je nižší. Z toho vyplývá, že metoda relaxace délky bude výhodnější hlavně pro menší kinematické struktury. To platí i pro porovnání s algoritmem CCD. Pro vyšší počet kinematických vazeb se stává CCD časově výhodnější. Na druhou stranu je více nestabilní a může se rozkmitat. Obecně jsou naměřené hodnoty spíše orientační, jelikož výsledný čas záleží do velké míry na počátečním postavení kinematické struktury. V případě metody relaxace délky na volbě aproximačního kinematického řetězce.

Tab. 8: Časové porovnání vybraných heuristických metod

Velikost planárního manipulátoru n	Metoda relaxace délky	FABRIK	CCD
	Ø Čas metody potřebný pro nalezení řešení [ms] s požadovanou přesností ± 0.01 mm		
2	19	21	76
3	77	79	77
4	164	164	124
5	94	94	68
6	222	225	100
7	361	361	155
8	353	353	159
9	379	379	129
10	430	430	196

5 Závěr

V rámci této disertační práce vznikly dva nové algoritmy pro řešení inverzní kinematické úlohy v robotice. Jedná se metodu relaxace úhlu a o metodu relaxace délky. Obě metody vychází z jednoduchého geometrického popisu bodů a vektorů v prostoru. Metoda relaxace úhlu je především určena pro řešení soustav lineárních rovnic, ale dá se nepřímo použít pro výpočet inverzní kinematické úlohy v robotice. Metoda relaxace délky je vyloženě heuristickou metodou pro řešení inverzní kinematické úlohy v robotice. Oba algoritmy byly diskutovány z hlediska jejich vlastností a schopností řešit předepsané úlohy. Jelikož problematika řešení soustav lineárních rovnic a řešení inverzní kinematické úlohy je v robotice běžnou inženýrskou disciplínou, našla metoda relaxace úhlu uplatnění v praktické aplikaci. Jedná se o časovou optimalizaci robotové trajektorie v rámci softwaru SKODA TOOL, který je vyvíjen pro kontrolu rozsáhlých robotických soustav ve svařovnách firmy ŠKODA AUTO a.s..

Při zkoumání vlastností metody relaxace úhlu v rámci problematiky řešení soustav lineárních rovnic bylo zjištěno, že se jedná o určitou iterační obdobu Mooreovy-Penroseovy pseudoinverze. Pseudoinverze je metoda přímá a relaxace úhlu metoda iterační. Přesto z hlediska dosažených výsledků vykazují obdobné chování. Jedná se především o tyto čtyři vlastnosti, které byly experimentálně ověřeny:

- pokud má soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ jedno řešení $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) = n$, tak metoda relaxace úhlu konverguje k tomuto řešení,
- pokud má soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ nekonečně mnoho řešení $h(\mathbf{A}) = h(\mathbf{A}|\mathbf{b}) < n$, tak se metoda relaxace úhlu blíží k řešení nejmenší v euklidovské normě $\|\mathbf{x}_{RLX}\|_e \approx \|\mathbf{x}\|_e \leq \|\mathbf{y}\|_e$,
- pokud soustava $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ nemá řešení $h(\mathbf{A}) < h(\mathbf{A}|\mathbf{b})$, tak se metoda relaxace úhlu blíží k řešení s nejmenším reziduem v euklidovské normě $\|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}_{RLX}\|_e \approx \|\mathbf{b} - \mathbf{A} \cdot \mathbf{x}\|_e \leq \|\mathbf{b} - \mathbf{A} \cdot \mathbf{y}\|_e$,
- výpočet funguje i pro obdélníkové matice.

Z hlediska možných řešení se metoda relaxace úhlu chová jako pseudoinverze, přesto její iterační základ přináší určité komplikace. Metoda relaxace úhlu vykazuje tak vysoký nárůst výpočetního času v závislosti na velikosti matice a číslu podmíněnosti matice soustavy, že se nedá uvažovat jako nová obecná metoda pro výpočet soustav lineárních rovnic. Toto bylo experimentálně ověřeno v rámci porovnání výpočetní rychlosti vybraných přímých a iteračních metod pro řešení soustav lineárních rovnic.

Navzdory špatným časovým výsledkům v rámci řešení soustav lineárních rovnic se ukázalo, že metoda relaxace úhlu může najít své uplatnění v oblasti robotiky. Při řešení inverzní kinematické úlohy v robotice jsou totiž kladeny úplně jiné požadavky na numerické metody. Při výpočtu inverzní

kinematické úlohy v robotice se pracuje převážně s omezenou velikostí matice soustavy, která se dynamicky mění v závislosti na postavení robota. Přitom je nutné předpokládat všechny možné podoby matice, např. singulární. Proto se při numerických výpočtech v robotice používá právě pseudoinverze, ačkoliv nepatří k těm nejrychlejším přímým metodám z hlediska výpočetního času. Omezená velikost matice a požadavky na nasazení pseudoinverze jsou věci, které podporují použití metody relaxace úhlu pro výpočet inverzní kinematické úlohy v robotice. Pro ověření vlastností metody relaxace úhlu byly vybrány tři numerické metody řešení inverzní kinematické úlohy v robotice, které bylo možné upravit do podoby soustavy lineárních rovnic. Jednalo se o Newtonovu metodu, inverzi Jacobiho matice a metodu Levenberg-Marquardt. Na praktické úloze s planárním manipulátorem a průmyslovým robotem KUKA KR210 R2700 EXTRA byla metoda relaxace úhlu odzkoušena a porovnána s chováním pseudoinverze. V rámci tohoto experimentu bylo zjištěno, že metodu relaxace úhlu lze použít pro numerické výpočty inverzní kinematické úlohy v robotice v kombinaci s metodou inverze Jacobiho matice a metodou Levenberg-Marquardt. S Newtonovou metodou nebyly výsledky až tak dobré. Experiment také ukázal, že mimo oblast singulárního postavení kinematické struktury jsou časové výsledky metody relaxace úhlu téměř srovnatelné s výsledky pseudoinverze. Zásadní rozdíly byly pozorovány pouze v oblasti singulárního postavení kinematické struktury. Pro větší diskretizační krok jsou v této oblasti časové výsledky metody relaxace úhlu horší než výsledky pseudoinverze. Při snižování diskretizačního kroku se potom časové výsledky metody relaxace úhlu zlepšují a u pseudoinverze zhoršují. Dokonce může být metoda relaxace úhlu rychlejší, než pseudoinverze. Snižování diskretizačního kroku vede u pseudoinverze ke ztrátě spojitosti a špatnému sledování trajektorie. Metoda relaxace úhlu byla proti tomuto chování více odolná.

Jak metoda relaxace úhlu, tak pseudoinverze, má především v oblasti singulárního postavení kinematické struktury své výhody, resp. nevýhody. Záleží na praktických požadavcích dané aplikace. V tomto ohledu je metoda relaxace úhlu novou alternativou, která může pomoci v situaci, kdy výsledky pseudoinverze nemusí být uspokojivé, např. při sledování předepsané trajektorie v oblasti singulárního postavení robota. Metoda relaxace úhlu nabízí ještě další vlastnosti, které se v oblasti robotiky dají využít. Metoda relaxace úhlu v rámci několika iteračních kroků velice rychle minimalizuje reziduum a tím dokáže nabídnout rychlý odhad řešení. Je zde i možnost kombinovat obě metody dohromady. Můžeme tak získat rychlý odhad řešení na základě metody relaxace úhlu, a ten zkorigovat přesným řešením získaným pomocí pseudoinverze. Významnou roli hraje struktura metody relaxace úhlu, která odpovídá struktuře zpětnovazebního regulátoru. To je příznivá podoba pro nasazení v rámci regulačního subsystému polohování robota. Metodu relaxace úhlu nelze z hlediska časových výsledků srovnávat s obecnými metodami řešení soustav lineárních rovnic, přesto v oblasti robotiky může nalézt své uplatnění. Díky možnosti rychlého odhadu řešení s vlastnostmi pseudoinverze ji lze z tohoto úhlu pohledu považovat také za rychlou.

Metoda relaxace délky vznikla jako variace metody relaxace úhlu. V disertační práci byla porovnána s vybranými heuristikami pro řešení inverzní kinematické úlohy v robotice CCD a FABRIK. Bylo zjištěno, že metoda relaxace délky odpovídá principiálně metodě FABRIK. Tím, že byla metoda relaxace délky testována nezávisle, jsou zde principiální rozdíly a některá vylepšení. Metoda relaxace délky v rámci algoritmu nepřesouvá koncové body kinematického řetězce, čímž šetří oproti FABRIK výpočetní čas. Tato časová úspora se projeví především pro malé kinematické struktury, kde není tak velký poměr počtu koncových bodů vůči všem relaxovaným. Jinak byly časové výsledky obou metod v rámci testování srovnatelné. Metoda CCD vykazovala lepší časové výsledky pro větší kinematické struktury. Zde se na druhou stranu stávala v některých případech nestabilní. Dále byla v této práci popsána aplikace metody relaxace délky pro paralelní kinematické struktury a možnost omezení pohybu kinematické vazby do předepsané roviny. Práce v tomto ohledu představuje novou myšlenku použití přídatné kinematické struktury. I v tomto se metoda relaxace délky liší od metody FABRIK.

Z hlediska dalšího výzkumu by bylo vhodné především u metody relaxace úhlu provést ověření a potvrzení experimentálně zjištěných vlastností odborníkem z oblasti matematiky. Chování metody relaxace úhlu bylo ověřeno na základě mnoha výpočtů typově různých soustav lineárních rovnic, ale kromě důkazu konvergence zde nebyly příčiny tohoto chování více diskutovány. Bylo by také zajímavé hledat další oblasti nasazení metody relaxace úhlu, především v oblasti řídicí techniky a regulace. Např. použití metody relaxace úhlu jako regulátoru pro MIMO systémy. Lineární optimalizace se používá v mnoha různých oborech. Tam, kde se pracuje s malými a dynamicky se měnícími soustavami, může metoda relaxace úhlu nalézt své uplatnění. Metoda relaxace úhlu se principiálně hodí k paralelizaci výpočtu, jelikož lze každou relaxaci úhlu počítat nezávisle. Pokud by byl počet paralelních vláken shodný s počtem hledaných neznámých, odpovídal by výpočetní čas relaxace matici o velikosti jedna. Bylo by potom zajímavé porovnat, jak by se metoda relaxace úhlu časově lišila oproti jiným používaným metodám.

Použitá literatura

- [1] V. H. John Lloyd. *Kinematics of common industrial robots*. vol. 4, no. 2, pp. 169-191, June 1988.
- [2] PRESS, William H. a Saul A. TEUKOLSKY. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. New York: Cambridge University Press, 2007. ISBN 0521880688
- [3] REKTORYS, Karel. *Přehled užití matematiky. II [Rektorys, 2000]*. 7. vyd. Praha: Prometheus, 2000. 874 s. ISBN 80-7196-181-72.
- [4] BUTENKO, Sergiy a P. M. PARDALOS. *Numerical methods and optimization: an introduction*. Chapman & Hall/CRC numerical analysis and scientific computing. ISBN 9781466577770.
- [5] HOUSEHOLDER, Alston Scott. *The theory of matrices in numerical analysis*. Dover ed. Mineola, N.Y.: Dover Publications, 2006. ISBN 0486449726.
- [6] *Numerical Methods in Matrix Computations*. Springer Verlag, 2014. ISBN 3319050885.
- [7] PYTLÍČEK, Jiří. *Lineární algebra a geometrie*. Praha : Česká technika - nakladatelství ČVUT, 2008. ISBN 978-80-01-04063-8. skripta FJFI ČVUT
- [8] BEČVÁŘ, J. *Lineární algebra*. 3. vyd. Praha: Matfyzpress, 2005. 435 s. ISBN 80-86732-57-6.
- [9] LAY, David C., Steven R. LAY a Judith. MCDONALD. *Linear algebra and its applications*. Fifth edition. ISBN 032198238x.
- [10] FRIEDBERG, Stephen H., Arnold J. INSEL a Lawrence E. SPENCE. *Linear algebra*. 3rd ed. Upper Saddle River, N.J.: Prentice Hall, c1997. ISBN 0132338599
- [11] LEWIS, Frank L., C. T. ABDALLAH a D. M. DAWSON. *Robot manipulator control: theory and practice*. 2nd ed., rev. and expanded. New York: Marcel Dekker, c2004. Control engineering (Marcel Dekker, Inc.). ISBN 0824740726
- [12] KOIVO, Antti J. *Fundamentals for control of robotic manipulators*. New York: Wiley, c1989. ISBN 0471857149.
- [13] J. Lander. *Making Kine more flexible*. Game Developer Magazine, vol.11, pp. 15-22, 1998.
- [14] D. Tolani, A. Goswami, and N. Badler. *Real-time inverse kinematics techniques for anthropomorphic limbs*. Graphical Models, Vol. 62, No. 5, pp. 353–388, 2000.
- [15] A. P. M. L.-W. Tsai. *Solving the Kinematics of the Most General SixandFive-Degree-of-Freedom Manipulators by Continuation Methods*. pp. 189-200, 01 Jun 1985.
- [16] ANGELES, Jorge. *Fundamentals of robotic mechanical systems: theory, methods, and algorithms*. 4th ed. Cham: Springer, 2014. ISBN 3319018507
- [17] Aristidou, A and Lasenby, J (2009) *Inverse kinematics: a review of existing techniques and introduction of a new fast iterative solver*. Technical Report. Cambridge University Engineering Department.
- [18] *Industrial Robotics: Theory, Modelling and Control*. Germany: Pro Literatur Verlag. ISBN 3-86611-285-8.
- [19] VASILYEV, I. A. a A. M. LYASHIN. *Analytical solution to inverse kinematic problem for 6-DOF robot-manipulator*. DOI: 10.1134/S0005117910100218. ISBN 10.1134/S0005117910100218. Dostupné z: <http://link.springer.com/10.1134/S0005117910100218>

- [20] SPONG, Mark W. a M. VIDYASAGAR. *Robot Dynamics and Control*. ISBN 978-0-471-61243-8
- [21] ARISTIDOU, Andreas a Joan LASENBY. *Inverse Kinematics Solutions Using Conformal Geometric Algebra*. DOI: 10.1007/978-0-85729-811-9_3. ISBN 10.1007/978-0-85729-811-9_3. Dostupné z: http://link.springer.com/10.1007/978-0-85729-811-9_3
- [22] CARBAJAL-ESPINOSA, O., F. IZAR-BONILLA, M. DIAZ-RODRIGUEZ a E. BAYRO-CORROCHANO. *Inverse kinematics of a 3 DOF parallel manipulator: A conformal geometric algebra approach*. DOI: 10.1109/HUMANOIDS.2016.7803360. ISBN 10.1109/HUMANOIDS.2016.7803360. Dostupné z: <http://ieeexplore.ieee.org/document/7803360/>
- [23] VACHARAKORNRAWUT, Nuttaprop, Teerawat THEPMANEE, Apinai RERKRATN a Sawai PONGSWATD. *Converting TCP to joints value of 6-DOF robot based on forward and inverse kinematic analysis*. DOI: 10.1109/ECTICon.2016.7561348. ISBN 10.1109/ECTICon.2016.7561348. Dostupné z: <http://ieeexplore.ieee.org/document/7561348/>
- [24] SHIHABUDHEEN, K V, G N PILLAI, Apinai RERKRATN a Sawai PONGSWATD. *Evolutionary fuzzy extreme learning machine for inverse kinematic modeling of robotic arms*. DOI: 10.1109/NATSYS.2015.7489105. ISBN 10.1109/NATSYS.2015.7489105. Dostupné také z: <http://ieeexplore.ieee.org/document/7489105/>
- [25] CAVDAR, Tugrul, M. MOHAMMAD a R. Alavi MILANI. *A New Heuristic Approach for Inverse Kinematics of Robot Arms*. DOI: 10.1166/asl.2013.4700. ISBN 10.1166/asl.2013.4700. Dostupné z: <http://openurl.ingenta.com/content/xref?genre=article>
- [26] AYYILDIZ, Mustafa a Kerim ÇETINKAYA. *Comparison of four different heuristic optimization algorithms for the inverse kinematics solution of a real 4-DOF serial robot manipulator*. DOI: 10.1007/s00521-015-1898-8. ISBN 10.1007/s00521-015-1898-8. Dostupné z: <http://link.springer.com/10.1007/s00521-015-1898-8>
- [27] POZNA, Claudiu Radu, Erno HORVATH a Janos HOLLOSI. *The inverse kinematics problem, a heuristical approach*. DOI: 10.1109/SAMI.2016.7423024. ISBN 10.1109/SAMI.2016.7423024. Dostupné z: <http://ieeexplore.ieee.org/document/7423024/>
- [28] SONG, Wei a Guang HU. *A Fast Inverse Kinematics Algorithm for Joint Animation*. DOI: 10.1016/j.proeng.2011.11.2655. ISBN 10.1016/j.proeng.2011.11.2655. Dostupné z: <http://linkinghub.elsevier.com/retrieve/pii/S187770581105507X>
- [29] ARISTIDOU, Andreas a Joan LASENBY. *FABRIK: A fast, iterative solver for the Inverse Kinematics problem*. DOI: 10.1016/j.gmod.2011.05.003. ISBN 10.1016/j.gmod.2011.05.003. Dostupné z: <http://linkinghub.elsevier.com/retrieve/pii/S1524070311000178>
- [30] *Types of Kinematic Joints in the Design of Machines* [online]. USA: YUCHAO, 2017. Dostupné z: https://www.yuchao.us/2017_09_13_archive.html
- [31] SMUTNÝ, Vladimír. *Kinematika robotů* [online]. Praha, 2012. Dostupné z: https://cw.fel.cvut.cz/old/_media/courses/a3b99ro/robotismutnycz.pdf. Přednáška. ČVUT.

- [32] *Euler angles* [online]. Autodesk, 2016. Dostupné z: <http://help.autodesk.com/view/MAYAUL/2016/ENU/?guid=GUID-CBD30A0A-1166-4076-A564-1ADC946A15F3>
- [33] HAYAT, Abdullah Aamir, Rajeevlochana G. CHITTAWADIGI, Arun Dayal UDAI a Subir K. SAHA. *Identification of Denavit-Hartenberg Parameters of an Industrial Robot. Proceedings of Conference on Advances In Robotics - AIR '13. New York, New York, USA: ACM Press, 2013, 2013, , 1-6. DOI: 10.1145/2506095.2506121. ISBN 9781450323475. Dostupné také z: <http://dl.acm.org/citation.cfm?doid=2506095.2506121>*
- [34] *Indiamart* [online]. Indije: IndiaMART InterMESH, 2019. Dostupné z: <https://www.indiamart.com/proddetail/cartesian-robot-2-3-4-axis-9082770748.html>
- [35] *Industrial robots* [online]. Japonsko: SCARA robots, 2019. Dostupné z: <https://global.yamahamotor.com/business/robot/lineup/ykxg/index.html>
- [36] *M-410iC/185* [online]. Japonsko: FANUC, 2019. Dostupné z: <https://www.fanuc.eu/cz/cs/roboty/str%C3%A1nka-filtru-robot%C5%AF/%C5%99ada-m-410/m-410ic-185>
- [37] *R12 5-Axis Articulated Robot Arm* [online]. Evropa: ST Robotics, 2019. Dostupné také z: <https://www.robotshop.com/en/st-robotics-r12-5-axis-articulated-robot-arm.html>
- [38] *KR210 R2700 EXTRA* [online]. Evropa: KUKA, 2019. Dostupné také z: <https://www.kuka.com/cs-cz/produkty,-slu%C5%BEby/robotick%C3%A9-syst%C3%A9my/pr%C5%AFmyslov%C3%A9-roboty/kr-quantec-extra>
- [39] *LBR IIWA* [online]. Evropa: KUKA, 2019. Dostupné také z: <https://www.kuka.com/cs-cz/produkty,-slu%C5%BEby/robotick%C3%A9-syst%C3%A9my/pr%C5%AFmyslov%C3%A9-roboty/lbr%C2%A0iiwa>
- [40] *8-Axis FaroArm* [online]. Evropa: FARO, 2019. Dostupné také z: <https://www.faro.com/news/breakthrough-faro-8-axis-faroarm-sets-new-standard/>
- [41] *How To Be A Career Robot Snake* [online]. USA: Work it daily, 2019. Dostupné také z: <https://www.workitdaily.com/career-robot-snake/>
- [42] *0406_CS_R33_Expert_VKRC_programování pro experty*. Augsburg, 2004.

Seznam publikací autora

- [1] ŠIDLOF P., DOARE O., CADOT O., ČEJKA J. (2011). Coherent turbulent structures in flow through the human vocal tract. Experimental Fluid Mechanics, Liberec, Czech Republic
- [2] CEJKA, Jan a Josef CERNOHORSKY. Optimization of robotic workplaces. Elektrotechnické listy. 2016. DOI: 10.1109/CarpathianCC.2016.7501083. ISBN 10.1109/CarpathianCC.2016.7501083. ISSN 2453-8981. Dostupné také z: <http://ieeexplore.ieee.org/document/7501083/>
- [3] CEJKA, Jan a Josef CERNOHORSKY. Inverse kinematic problem solved by new heuristic algorithm length relaxation method. DOI: 10.1109/CarpathianCC.2017.7970453. ISBN 10.1109/CarpathianCC.2017.7970453. Dostupné také z: <http://ieeexplore.ieee.org/document/7970453/>
- [4] CEJKA, Jan a Josef CERNOHORSKY. Řešení inverzní kinematické úlohy pomocí nové heuristické metody relaxace délky. Elektrotechnické listy. 2016. ISSN 2453-8981. Dostupné také z: http://elektrotechnickelisty.eu/casopis/rocnik_I/cislo_1_2016.html