TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Rychlé heuristické metody numerického řešení úlohy inverzní kinematiky

Autoreferát 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.

Liberec 2019

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 robotu 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

$\mathbb{R}, \mathbb{C}, \mathbb{N}$	obor reálných, komplexních, přirozených čísel
$\mathbb{R}^n$	aritmetický vektorový prostor tvořený n-ticemi reálných čísel
V	vektorový prostor
$\boldsymbol{x}$	vektor
$\boldsymbol{x}^R$	relaxovaný vektor
$\widehat{\boldsymbol{x}}$	normovaný vektor
a, b, ..	skalár
$\boldsymbol{o}$	nulový vektor
$\boldsymbol{O}$	nulová matice
$\boldsymbol{A}$	matice soustavy
$\boldsymbol{P}$	matice nevlastního řešení soustavy
$\boldsymbol{r}$	reziduum
$\boldsymbol{p}$	sloupec matice nevlastního řešení soustavy
$\mathbb{E}_n$	eukleidovský prostor konečné dimenze n
$P_1, P_2, \ldots$	body
$\oplus$ +	sčítání vektorů
$\odot \cdot$	násobení vektorů
$\alpha, \beta, \ldots$	číslo z tělesa
M	množina
$\kappa(\boldsymbol{A})$	číslo podmíněnosti matice
$\boldsymbol{A^{-1}}$	inverze matice
$\boldsymbol{A^{+}}$	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

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. 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. 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 syntaxi 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.

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.

S postupem času se stal software SKODA TOOL platformou pro různé vědecké experimenty. Jak již bylo jednou řečeno, rozsáhlé robotové soustavy 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.

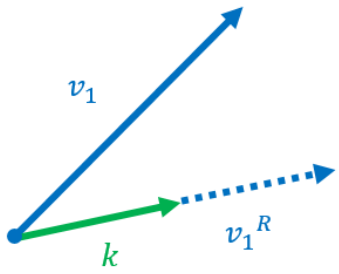
Hlavní cíle disertační práce jsou:

- 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.

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 $\boldsymbol{v_1}$ a vektor $\boldsymbol{k}$ (viz obr. 1). Vektor $\boldsymbol{k}$ nám ukazuje směr, do kterého chceme vektor $\boldsymbol{v_1}$ otočit. Vznikne nám tak nový vektor $\boldsymbol{v_1}^R$, který je lineárně závislý s vektorem $\boldsymbol{k}$ a má velikost vektoru $\boldsymbol{v_1}$.



Obr. 1: Relaxace úhlu

Definice 2.1. Relaxaci úhlu definujeme jako otočení vektoru $\boldsymbol{v}_1$ do směru $\boldsymbol{k}$ (1). Tento jednoduchý výpočet bude základním stavebním kamenem celého algoritmu.

$$\boldsymbol{v}_1{}^R = \frac{\boldsymbol{k}}{\|\boldsymbol{k}\|_e}\|\boldsymbol{v}_1\|_e \tag{1}$$

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 $x_1, x_2, x_3, \ldots, x_n$ nazveme soustavu (2) kde $a_{11}, \ldots, a_{mn}$ a $b_1, \ldots, b_m \in \mathbb{R}$.

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

Řešením soustavy (2) se nazývá každá uspořádaná n-tice $(x_1, x_2, x_3, \ldots, x_n)^T \in \mathbb{R}$, tj. n-členný vektor, který splňuje všechny rovnice soustavy (2).

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

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

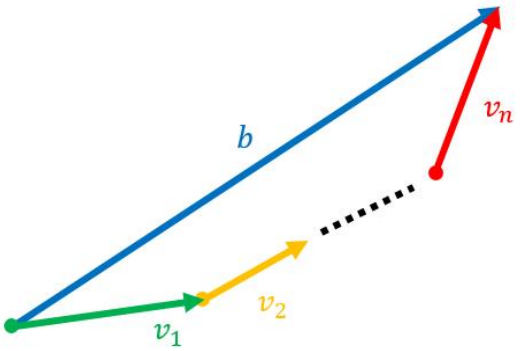
V našem případě budeme vycházet z ještě jiné možné podoby zápisu. Na soustavu budeme nahlížet z geometrického hlediska. Vektor pravých stran $\boldsymbol{b}$ je roven lineární kombinaci sloupců matice soustavy $\boldsymbol{A}$ (3).

$$\begin{aligned} \boldsymbol{A}_1 \cdot x_1 + \boldsymbol{A}_2 \cdot x_2 + \cdots \boldsymbol{A}_n \cdot x_n &= \boldsymbol{b} \\ \begin{pmatrix} a_{11} \\ \vdots \\ a_{m1} \end{pmatrix} \cdot x_1 + \begin{pmatrix} a_{12} \\ \vdots \\ a_{m2} \end{pmatrix} \cdot x_2 + \cdots \begin{pmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{pmatrix} \cdot x_n &= \boldsymbol{b} \end{aligned} \tag{3}$$

Tento zápis můžeme přepsat jako součet vektorů $\boldsymbol{v}_1$ až $\boldsymbol{v}_n$ (4). Vektory $\boldsymbol{v}_1$ až $\boldsymbol{v}_n$ jsou lineárně závislé se sloupci matice soustavy $\boldsymbol{A}$.

$$\begin{aligned} \boldsymbol{v}_1 + \boldsymbol{v}_2 + \cdots + \boldsymbol{v}_n &= \boldsymbol{b} \\ \boldsymbol{v}_1 = \boldsymbol{A}_1 \cdot x_1, \boldsymbol{v}_2 = \boldsymbol{A}_2 \cdot x_2, \ldots, \boldsymbol{v}_n &= \boldsymbol{A}_n \cdot x_n \end{aligned} \tag{4}$$

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 2. vidíme geometrickou interpretaci soustavy lineárních rovnic pomocí vektorů.



Obr. 2: Geometrická interpretace soustavy lineárních rovnic pomocí vektorů

Obecně můžeme vektor pravých stran $\boldsymbol{b}$ sestavit z libovolného počtu vektorů, které ovšem nemusí odpovídat soustavě lineárních rovnic, tzn. vektory nebudou lineárně závislé se sloupci matice soustavy $\boldsymbol{A}$. 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 $\boldsymbol{v}_1$ až $\boldsymbol{v}_n$, které jsou v součtu rovny vektoru pravých stran $\boldsymbol{b}$, ale nemusí být lineárně závislé se sloupci matice soustavy $\boldsymbol{A}$. Přičemž $\boldsymbol{X}_1$, $\boldsymbol{X}_2$,..., $\boldsymbol{X}_n$ jsou diagonální matice, kde diagonální složky nejsou stejně velké (5).

$$\begin{gathered} \boldsymbol{v}_1 + \boldsymbol{v}_2 + \cdots + \boldsymbol{v}_n = \boldsymbol{b} \\ \boldsymbol{X}_i = \begin{pmatrix} x_{i1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x_{im} \end{pmatrix}_i \\ \boldsymbol{v}_1 = \boldsymbol{X}_1 \cdot \boldsymbol{A}_1, \boldsymbol{v}_2 = \boldsymbol{X}_2 \cdot \boldsymbol{A}_2, .. , \boldsymbol{v}_n = \boldsymbol{X}_n \cdot \boldsymbol{A}_n \end{gathered} \tag{5}$$

Definice 2.4. Vlastním řešením soustavy budeme nazývat vektory, které jsou v součtu rovny vektoru pravých stran $\boldsymbol{b}$ a zároveň jsou lineárně závislé se sloupci matice soustavy $\boldsymbol{A}$. Přičemž x_1, x_2,..., x_n jsou skaláry (6).

$$\begin{gathered} \boldsymbol{v}_1 + \boldsymbol{v}_2 + \cdots + \boldsymbol{v}_n = \boldsymbol{b} \\ \boldsymbol{v}_1 = \boldsymbol{A}_1 \cdot x_1, \boldsymbol{v}_2 = \boldsymbol{A}_2 \cdot x_2, .. , \boldsymbol{v}_n = \boldsymbol{A}_n \cdot x_n \end{gathered} \tag{6}$$

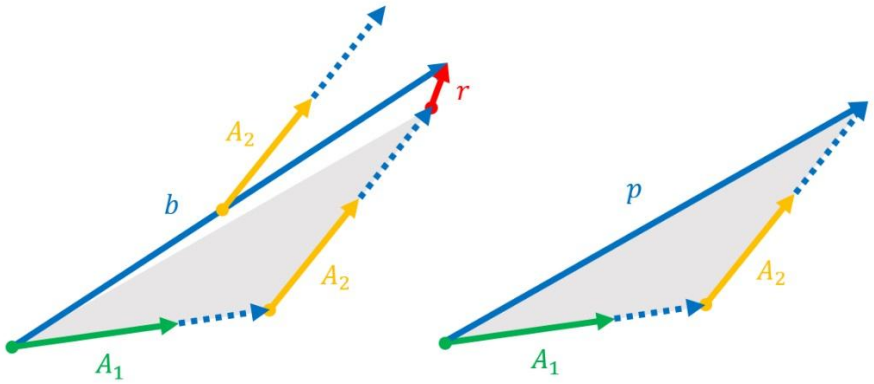
Nalézt vlastní řešení soustavy je náš cíl, ale zároveň určit, nebo odhadnout x_1, x_2,..., x_n je složité. Budeme proto postupovat tak, že nejprve nalezneme nevlastní řešení soustavy. To můžeme udělat velice jednoduše tak, že rozdělíme vektor $\boldsymbol{b}$ na několik stejných částí podle počtu hledaných neznámých n. Je jasné, že $\boldsymbol{v}_{10}$ až $\boldsymbol{v}_{n0}$ nemusí být lineárně závislé se sloupci matice soustavy $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$ (7).

$$\begin{gathered} \boldsymbol{v}_{10} + \boldsymbol{v}_{20} + \cdots + \boldsymbol{v}_{n0} = \boldsymbol{b} \\ \boldsymbol{v}_{10} = \frac{\boldsymbol{b}}{n}, \boldsymbol{v}_{20} = \frac{\boldsymbol{b}}{n}, .. , \boldsymbol{v}_{n0} = \frac{\boldsymbol{b}}{n} \end{gathered} \tag{7}$$

Následně musíme převést nevlastní řešení soustavy na vlastní řešení soustavy. Právě k tomuto převodu použijeme relaxaci úhlu (1) z definice 2.1. Tento krok je esenciálním prvkem celé metody. Z lineárně nezávislých vektorů $\boldsymbol{v}_{10}$ až $\boldsymbol{v}_{n0}$ učiníme vektory lineárně závislé k $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$ i za cenu toho, že nedosáhneme přesného řešení soustavy a zbyde nám odchylka, kterou nazveme reziduum $\boldsymbol{r}_k$ (8). Relaxací úhlu se převede matice $\boldsymbol{X}_i$ do skalární podoby x_i.

$$\pm \frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e}\|\boldsymbol{v}_{10}\|_e \pm \frac{\boldsymbol{A}_2}{\|\boldsymbol{A}_2\|_e}\|\boldsymbol{v}_{20}\|_e \pm \cdots \pm \frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e}\|\boldsymbol{v}_{n0}\|_e + \boldsymbol{r}_0 = \boldsymbol{b}$$
$$\pm {\boldsymbol{v}_{10}}^R \pm {\boldsymbol{v}_{20}}^R \pm \cdots \pm {\boldsymbol{v}_{n0}}^R + \boldsymbol{r}_0 = \boldsymbol{b} \tag{8}$$
$$\boldsymbol{p}_0 = \sum_{i=1}^{n} \pm {\boldsymbol{v}_{i0}}^R$$
$$\boldsymbol{p}_0 + \boldsymbol{r}_0 = \boldsymbol{b}$$

Příklad převodu pro $n = 2$ z nevlastního řešení soustavy na vlastní řešení soustavy je graficky znázorněn na obrázku 3.



Obr. 3: Geometrická intepretace převodu nevlastního řešení soustavu na řešení soustavy vlastní

Obecně neplatí, že se vektory musí pro co nejmenší reziduum $\boldsymbol{r}_k$ 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 $\|\boldsymbol{v}_{nk}\|_e$, čímž ztrácíme informaci o orientaci vektoru $\boldsymbol{b}$. Pokud by vektor $\boldsymbol{b}$ byl záporný a orientace některého z vektorů $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$ 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 $\boldsymbol{b}$.

Pokud bude znaménko kladné, platí nerovnice 9.

$$\left\|\frac{\boldsymbol{b}}{n} - \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e < \left\|\frac{\boldsymbol{b}}{n} + \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e \tag{9}$$

Pokud bude znaménko záporné, platí nerovnice 10.

$$\left\|\frac{\boldsymbol{b}}{n} - \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e > \left\|\frac{\boldsymbol{b}}{n} + \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e \tag{10}$$

Rovnost by odpovídala pouze nulovému vektoru $\boldsymbol{b}$.

Tento zápis se dá vyjádřit pomocí znaménkové funkce signum (11)

$$sgn\left(\left\|\frac{\boldsymbol{b}}{n} + \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e - \left\|\frac{\boldsymbol{b}}{n} - \frac{\boldsymbol{A}_i}{\|\boldsymbol{A}_i\|_e}\|\boldsymbol{v}_{i0}\|_e\right\|_e\right)_n \tag{11}$$

V druhém kroku algoritmu prohlásíme reziduum $\boldsymbol{r}_0$ novým hledaným vektorem pravých stran $\boldsymbol{b}_0 = \boldsymbol{r}_0$ a provedeme identické operace. Nejprve nalezneme nevlastní řešení soustavy (12).

$$\boldsymbol{v}_{11} + \boldsymbol{v}_{21} + \cdots + \boldsymbol{v}_{n1} = \boldsymbol{b}_0$$
$$\boldsymbol{v}_{11} = \frac{\boldsymbol{b}_0}{n}, \boldsymbol{v}_{21} = \frac{\boldsymbol{b}_0}{n}, \ldots, \boldsymbol{v}_{n1} = \frac{\boldsymbol{b}_0}{n} \tag{12}$$

Následně převedeme vektory na vlastní řešení soustavy pomocí relaxace úhlu (13).

$$\begin{gathered} \pm \frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e}\|\boldsymbol{v}_{11}\|_e \pm \frac{\boldsymbol{A}_2}{\|\boldsymbol{A}_2\|_e}\|\boldsymbol{v}_{21}\|_e \pm \cdots \pm \frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e}\|\boldsymbol{v}_{n1}\|_e + \boldsymbol{r}_1 = \boldsymbol{b}_0 \\ \pm {\boldsymbol{v}_{11}}^R \pm {\boldsymbol{v}_{21}}^R \pm \cdots \pm {\boldsymbol{v}_{n1}}^R + \boldsymbol{r}_1 = \boldsymbol{b}_0 \\ \boldsymbol{p}_1 = \sum_{i=1}^{n} \pm {\boldsymbol{v}_{i1}}^R \\ \boldsymbol{p}_1 + \boldsymbol{r}_1 = \boldsymbol{b}_0 = \boldsymbol{r}_0 \end{gathered} \tag{13}$$

V každém kroku algoritmu se velikost rezidua $\|\boldsymbol{r}_k\|_e$ zmenšuje, až se začne infinitezimálně blížit nulovému vektoru $\boldsymbol{o}$. Tento předpoklad si později dokážeme. Celkově pak platí, že součet všech vektorů $\boldsymbol{p}_i$ a posledního rezidua $\boldsymbol{r}_k$ je roven vektoru pravých stran $\boldsymbol{b}$ (14).

$$\boldsymbol{b} = \sum_{i=0}^{k} \boldsymbol{p}_i + \boldsymbol{b}_k = \sum_{i=0}^{k} \boldsymbol{p}_i + \boldsymbol{r}_k \tag{14}$$

Abychom mohli vyjádřit hledané neznámé x_1, x_2 až x_n, musíme sečíst lineárně závislé vektory k jednotlivým sloupcům matice soustavy $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$ z každého kroku iterace. Soustavu rovnic 15 převedeme na rovnici 16. Vektory $\boldsymbol{v}_{1c}$ až $\boldsymbol{v}_{nc}$ jsou lineárně závislé ke sloupcům matice soustavy $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$.

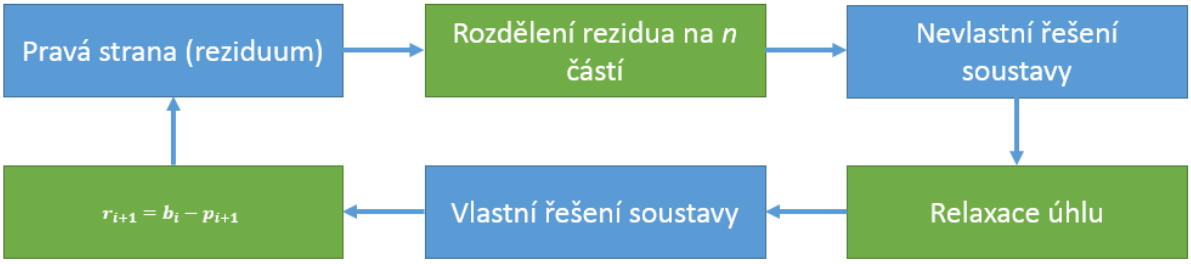
$$\begin{gathered} \pm {\boldsymbol{v}_{10}}^R \pm {\boldsymbol{v}_{20}}^R \pm \cdots \pm {\boldsymbol{v}_{n0}}^R = \boldsymbol{p}_0 \\ \pm {\boldsymbol{v}_{11}}^R \pm {\boldsymbol{v}_{21}}^R \pm \cdots \pm {\boldsymbol{v}_{n1}}^R = \boldsymbol{p}_1 \\ \vdots \\ \pm {\boldsymbol{v}_{1k}}^R \pm {\boldsymbol{v}_{2k}}^R \pm \cdots \pm {\boldsymbol{v}_{nk}}^R = \boldsymbol{p_k} \end{gathered} \tag{15}$$

$$\begin{gathered} \sum_{i=0}^{k} \pm {\boldsymbol{v}_{1i}}^R + \sum_{i=0}^{k} \pm {\boldsymbol{v}_{2i}}^R + \cdots + \sum_{i=0}^{k} \pm {\boldsymbol{v}_{ni}}^R = \sum_{i=0}^{k} \boldsymbol{p}_i = \boldsymbol{b} - \boldsymbol{r}_k \\ \boldsymbol{v}_{1c} + \boldsymbol{v}_{1c} + \cdots + \boldsymbol{v}_{nc} = \boldsymbol{p}_c = \boldsymbol{b} - \boldsymbol{r}_k \end{gathered} \tag{16}$$

Posledním krokem je vyjádření hledaných neznámých x_1, x_2 až x_n na základě rovnice 6 a 16. Jelikož x_1, x_2 až x_n je po relaxaci úhlu skalární hodnota, stačí vydělit libovolný prvek lineárně závislého vektoru $\boldsymbol{v}_{1c}$ až $\boldsymbol{v}_{nc}$ odpovídajícím prvkem z vektoru $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$ (17). Musíme si dát pozor na dělení nulou a pro výpočet vzít nenulový prvek z vektoru $\boldsymbol{A}_1$ až $\boldsymbol{A}_n$. Matice soustavy $\boldsymbol{A}$ může totiž obsahovat nulové prvky. Přesnost řešení odpovídá velikosti rezidua $\boldsymbol{r}_k$. Pokud je $\boldsymbol{r}_k = \boldsymbol{o}$, dosáhli jsme přesného řešení, jelikož $\boldsymbol{p}_c = \boldsymbol{b} - \boldsymbol{r}_k$.

$$x_1 = \frac{\boldsymbol{v}_{1cm}}{\boldsymbol{A}_{1m}}, x_2 = \frac{\boldsymbol{v}_{2cm}}{\boldsymbol{A}_{2m}}, \ldots, x_n = \frac{\boldsymbol{v}_{ncm}}{\boldsymbol{A}_{nm}} \tag{17}$$

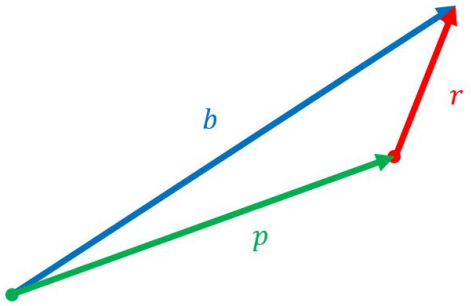
Pro názornost vyjádříme algoritmus blokově pomocí základních operací viz obrázek 4.



Obr. 4: 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 $\|\boldsymbol{r}\|_e$ se v každém kroku metody zmenšuje. Budeme vycházet z trojúhelníkové nerovnosti (obr. 5)(18).



Obr. 5: Trojúhelníková nerovnost

$$\begin{aligned} \|\boldsymbol{p}+\boldsymbol{r}\|_e &\le \|\boldsymbol{p}\|_e + \|\boldsymbol{r}\|_e \\ \|\boldsymbol{b}\|_e &\le \|\boldsymbol{p}\|_e + \|\boldsymbol{r}\|_e \\ \|\boldsymbol{b}\|_e - \|\boldsymbol{p}\|_e &\le \|\boldsymbol{r}\|_e \end{aligned} \tag{18}$$

Za vektor $\boldsymbol{p}$ dosadíme vztah pro relaxaci úhlu (19)

$$\|\boldsymbol{b}\|_e - \left\| \pm \frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e}\|\boldsymbol{v}_1\|_e \pm \frac{\boldsymbol{A}_2}{\|\boldsymbol{A}_2\|_e}\|\boldsymbol{v}_2\|_e \pm \cdots \pm \frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e}\|\boldsymbol{v}_n\|_e \right\| \le \|\boldsymbol{r}\|_e \tag{19}$$

Pro nevlastní řešení soustavy jsme volili $\boldsymbol{v}_1$ až $\boldsymbol{v}_n = \frac{\boldsymbol{b}}{n}$ (20). Obecně lze volit i jiné možné rozdělení vektoru $\boldsymbol{b}$ na n různých částí. Pozn.: volba rozdělení má vliv na rychlost konvergence a stabilitu řešení metody. Pro $\frac{\boldsymbol{b}}{n}$ bude metoda konvergovat vždy.

$$\|\boldsymbol{b}\|_e - \left\| \pm \frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e}\left\|\frac{\boldsymbol{b}}{n}\right\|_e \pm \frac{\boldsymbol{A}_2}{\|\boldsymbol{A}_2\|_e}\left\|\frac{\boldsymbol{b}}{n}\right\|_e \pm \cdots \pm \frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e}\left\|\frac{\boldsymbol{b}}{n}\right\|_e \right\| \le \|\boldsymbol{r}\|_e \tag{20}$$

Následně celou nerovnici upravíme a vytkneme konstanty (21).

$$\begin{aligned} \|\boldsymbol{b}\|_e - \left\|\frac{\boldsymbol{b}}{n}\right\|_e \cdot \left\| \pm \frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e} \pm \frac{\boldsymbol{A}_2}{\|\boldsymbol{A}_2\|_e} \pm \cdots \pm \frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e} \right\| &\le \|\boldsymbol{r}\|_e \\ \|\boldsymbol{b}\|_e - \frac{1}{n}\|\boldsymbol{b}\|_e \cdot \left\| \pm \widehat{\boldsymbol{A}}_1 \pm \widehat{\boldsymbol{A}}_2 \pm \cdots \pm \widehat{\boldsymbol{A}}_n \right\| &\le \|\boldsymbol{r}\|_e \end{aligned} \tag{21}$$

Jelikož vektory $\frac{\boldsymbol{A}_1}{\|\boldsymbol{A}_1\|_e} = \widehat{\boldsymbol{A}}_1$ až $\frac{\boldsymbol{A}_n}{\|\boldsymbol{A}_n\|_e} = \widehat{\boldsymbol{A}}_n$ jsou normované, může velikost jednoho z prvků $\widehat{\boldsymbol{A}}_1$ až $\widehat{\boldsymbol{A}}_n$ dosáhnout maximální hodnoty ± 1, např. $\widehat{\boldsymbol{A}}_i = (1,0)^T$, $\widehat{\boldsymbol{A}}_i = (-1,0)^T$, $\widehat{\boldsymbol{A}}_i = (0,-1)^T$ nebo $\widehat{\boldsymbol{A}}_i = (0,1)^T$. Z tohoto důvodu se celková velikost součtu prvků na pozici j normovaných vektorů $\widehat{\boldsymbol{A}}_1$ až $\widehat{\boldsymbol{A}}_n$ bude pohybovat v maximálním intervalu $\langle 0, \pm \mathsf{n} \rangle_j$. 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 $\widehat{\boldsymbol{A}}_1$ až $\widehat{\boldsymbol{A}}_n$ stejná, můžeme dosáhnout maximálního součtu $\pm \mathsf{n}$ (22).

$$\pm \widehat{\boldsymbol{A}}_{1j} \pm \widehat{\boldsymbol{A}}_{2j} \pm \cdots \pm \widehat{\boldsymbol{A}}_{nj} \in \langle 0, \pm \mathsf{n} \rangle_j \quad j = 1,2,\ldots,m \tag{22}$$

Celou nerovnici vyjádříme zjednodušeným vztahem 23. Přičemž výstupem z normy $\|(\langle 0, \pm \mathsf{n} \rangle_1, \langle 0, \pm \mathsf{n} \rangle_2, \ldots, \langle 0, \pm \mathsf{n} \rangle_m)^T\|_e$ musí být opět číslo z intervalu $\langle 0, \mathsf{n} \rangle$.

$$\|\boldsymbol{b}\|_e - \frac{1}{n}\|\boldsymbol{b}\|_e \cdot \|(\langle 0, \pm \mathsf{n} \rangle_1, \langle 0, \pm \mathsf{n} \rangle_2, \ldots, \langle 0, \pm \mathsf{n} \rangle_m)^T\|_e \leq \|\boldsymbol{r}\|_e \tag{23}$$

Řešení se bude pohybovat mezi dvěma extrémy. První extrém odpovídá vzájemnému odečtení vektorů. Potom z vektoru $\boldsymbol{b}$ žádný vektor neodečteme a reziduum $\boldsymbol{r}$ bude odpovídat vektoru $\boldsymbol{b}$ (24).

$$\begin{aligned} \|\boldsymbol{b}\|_e - \frac{1}{n}\|\boldsymbol{b}\|_e \cdot \|\boldsymbol{o}\|_e &\leq \|\boldsymbol{r}\|_e \\ \|\boldsymbol{b}\|_e &= \|\boldsymbol{r}\|_e \end{aligned} \tag{24}$$

Druhý extrém odpovídá situaci, kdy dosáhneme maximálního součtu. Například v případě prvního prvku je velikost normy $\|(\pm \mathsf{n}, 0, \cdots, 0)^T\|_e = n$. Reziduum $\boldsymbol{r}$ pak bude větší, nebo rovno nule (25).

$$\begin{aligned} \|\boldsymbol{b}\|_e - \frac{1}{n}\|\boldsymbol{b}\|_e \cdot \|(\pm \mathsf{n}, 0, \cdots, 0)^T\|_e &\leq \|\boldsymbol{r}\|_e \\ \|\boldsymbol{b}\|_e - \frac{1}{n}\|\boldsymbol{b}\|_e \cdot n &\leq \|\boldsymbol{r}\|_e \\ 0 &\leq \|\boldsymbol{r}\|_e \end{aligned} \tag{25}$$

Nyní můžeme konstatovat, že velikost rezidua $\|\boldsymbol{r}\|_e$ se bude vždy pohybovat v intervalu $\langle 0, \|\boldsymbol{b}\|_e \rangle$, jelikož dle nerovnice 24 a 25 se musí pohybovat mezi těmito extrémy. To znamená, že se $\|\boldsymbol{r}\|_e$ musí v každém kroku metody zmenšovat až na mezní situaci $\|\boldsymbol{r}\|_e = \|\boldsymbol{b}\|_e$. Je splněno tzv. D'Alembertovo kritérium konvergence. Mezní situace $\|\boldsymbol{r}\|_e = \|\boldsymbol{b}\|_e$ 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 $\boldsymbol{b}$, resp. $\boldsymbol{b}_0$, $\boldsymbol{b}_1$, atd. na stejné části, ale pouze jeho náhodnou část z intervalu $(0,1)$ (26). Tento krok nám odstraní problémy pro mezní situace a také zaručí konvergenci algoritmu k nule. Zavedení náhodného prvku může také zlepšit rychlost konvergence.

$$\boldsymbol{v}_{10} = \sigma \cdot \frac{\boldsymbol{b}}{n}, \boldsymbol{v}_{20} = \sigma \cdot \frac{\boldsymbol{b}}{n}, \ldots, \boldsymbol{v}_{n0} = \sigma \cdot \frac{\boldsymbol{b}}{n} \tag{26}$$

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 (viz obr. 6).

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

$$\boldsymbol{K} = \left(\widehat{\boldsymbol{A}}_1\ \widehat{\boldsymbol{A}}_2\ \ldots \widehat{\boldsymbol{A}}_n\right)$$

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

While (*není podmínka zastavení*)

$$\boldsymbol{V}_k = \left(\frac{\boldsymbol{r}_k}{n}\ \frac{\boldsymbol{r}_k}{n} \ldots \frac{\boldsymbol{r}_k}{n}\right)$$

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

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

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

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

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

End

$$\boldsymbol{x} = \left(\frac{\boldsymbol{P}_{1j}}{\boldsymbol{A}_{1j}}\ \frac{\boldsymbol{P}_{2j}}{\boldsymbol{A}_{2j}} \ldots \frac{\boldsymbol{P}_{nj}}{\boldsymbol{A}_{nj}}\right)^T$$

Obr. 6 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\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\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. 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 **A**. Pseudoinverze je metoda, která je oproti ostatním přímým metodám pomalejší, ale dostatečně robustní, aby si poradila se špatně podmíněnými i obdélníkovými maticemi soustavy **A**, což je vidět na výsledcích našeho prvního experimentu (viz tab. 1).

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. 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.

Tab. 1: Porovnání konvergence vybraných numerických metod

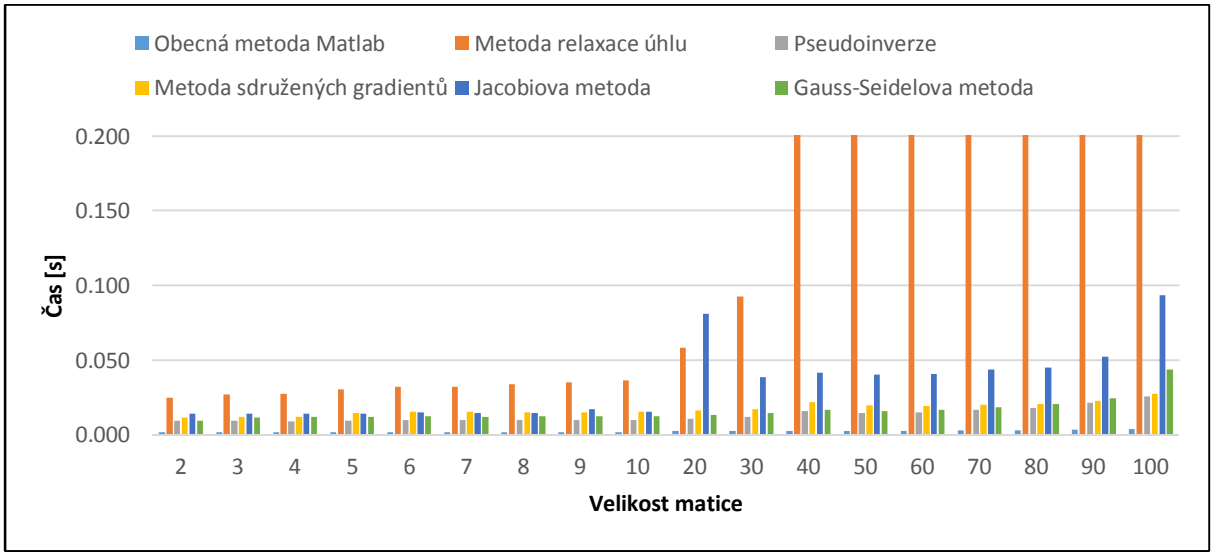
Vlastnosti	**Matice soustavy**										**Rozměr**
	Involutorní	Cauchyho	Tridiagonální (s nuly na diagonále)	Hankelova	Horní trapezoidální	Přeurčená	Nedourčená	Lehmerova	Dorrova	Poissonova	
Pozitivně definitní	Ne	Ano	Ne	Ano	Ano	Ne	Ne	Ano	Ano	Ano	m=n
Diagonálně dominantní	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ano	Ano	
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	
Obdelníková	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

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 $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) = n$. Iterační metody jsme zastavili, pokud norma rezidua $\|\boldsymbol{r}\|_e$ 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 7 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 podmíněnosti	Obecná metoda x=A\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

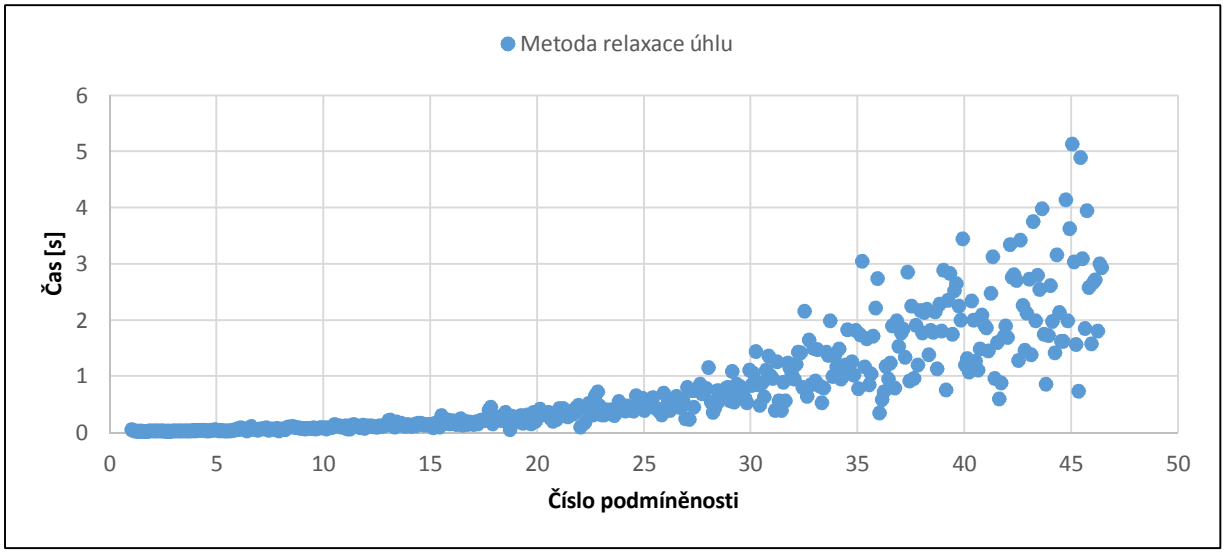


Obr. 7 Čas potřebný pro vyřešení soustavy lineárních rovnic vybraných numerických metod

Dále jsme testovali vliv čísla podmíněnosti matice soustavy (27) na rychlost konvergence metody relaxace úhlu (viz. obr. 18). Pro stejně velkou soustavu lineárních rovnic $n = 10$ jsme generovali matice soustavy se stále větším číslem podmíněnosti. Zastavení metody bylo opět podmíněno normou rezidua $\|\boldsymbol{r}\|_e < 0.01$, nebo dosažením počtu iterací 20 000.

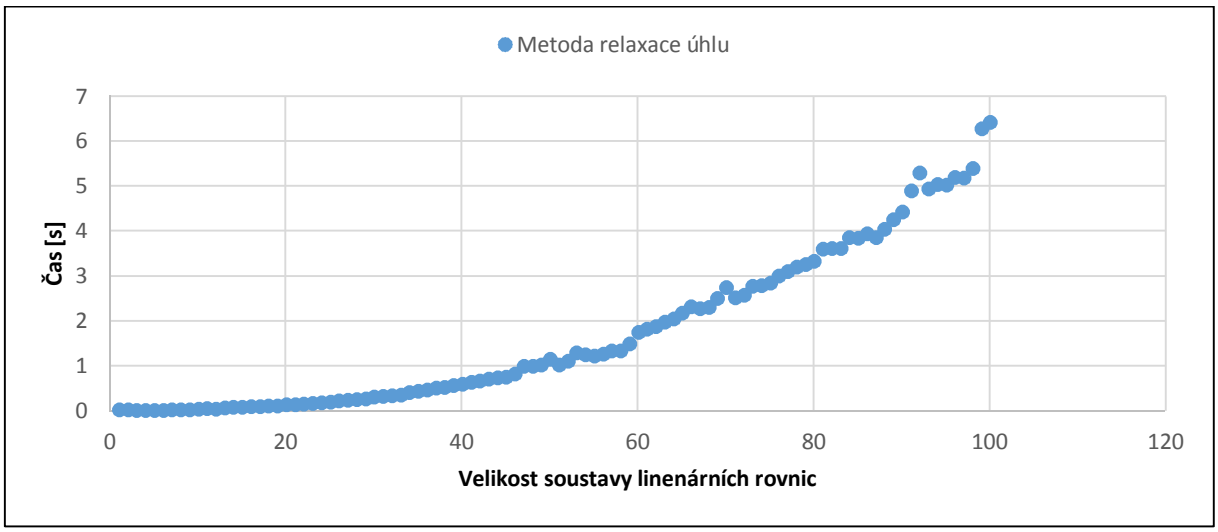
Číslo podmíněnosti se určuje v softwaru MATLAB dle vztahu 27. Pseudoinverze umožnuje stanovit číslo podmíněnosti i pro singulární matice, které nejsou invertovatelné.

$$\kappa(\boldsymbol{A}) = \|\boldsymbol{A}\|_e \cdot \|\boldsymbol{A}^+\|_e \tag{27}$$



Obr. 8 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 9.



Obr. 9: Vliv velikosti soustavy lineárních rovnic na rychlosti konvergence metody relaxace úhlu, κ(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 přeurčená $m > n$, 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. $\|\boldsymbol{r}\|_e < 0.01$, nebo 20 000 iterací.

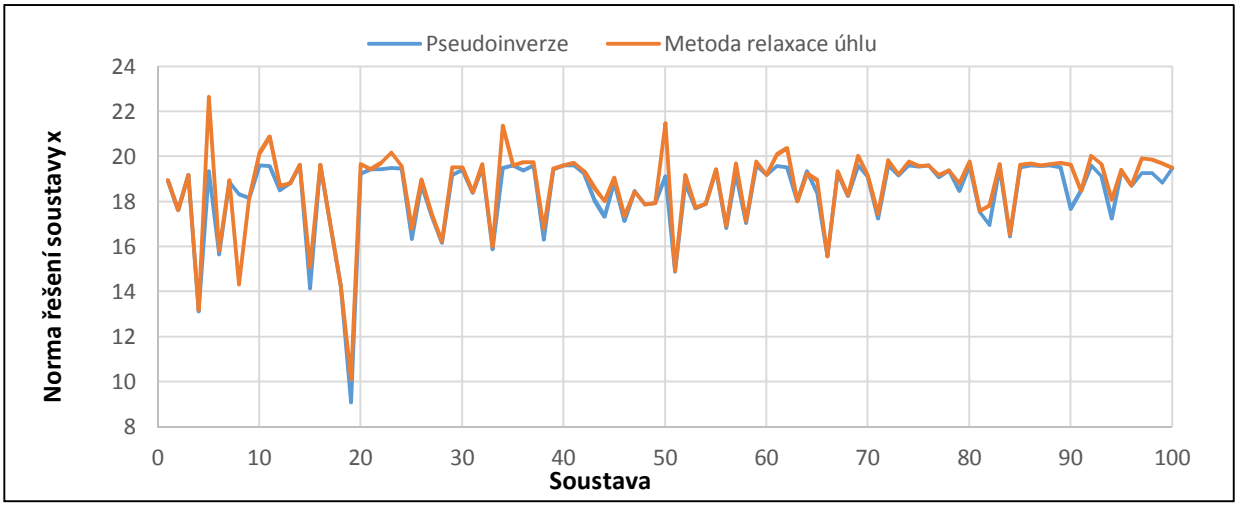
Tab. 3: Časové porovnání vybraných numerických metod pro obdélníkové matice

Matice m<n			Ø Čas [s]			Matice m>n			Ø Čas [s]		
m	n	Číslo podmíněnosti	Obecná metoda x=A\b Matlab	Metoda relaxace úhlu	Pseudoinverze	m	n	Číslo podmíněnosti	Obecná metoda x=A\b Matlab	Metoda relaxace úhlu	Pseudoinverze
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 podmíněnosti 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. 7). 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..

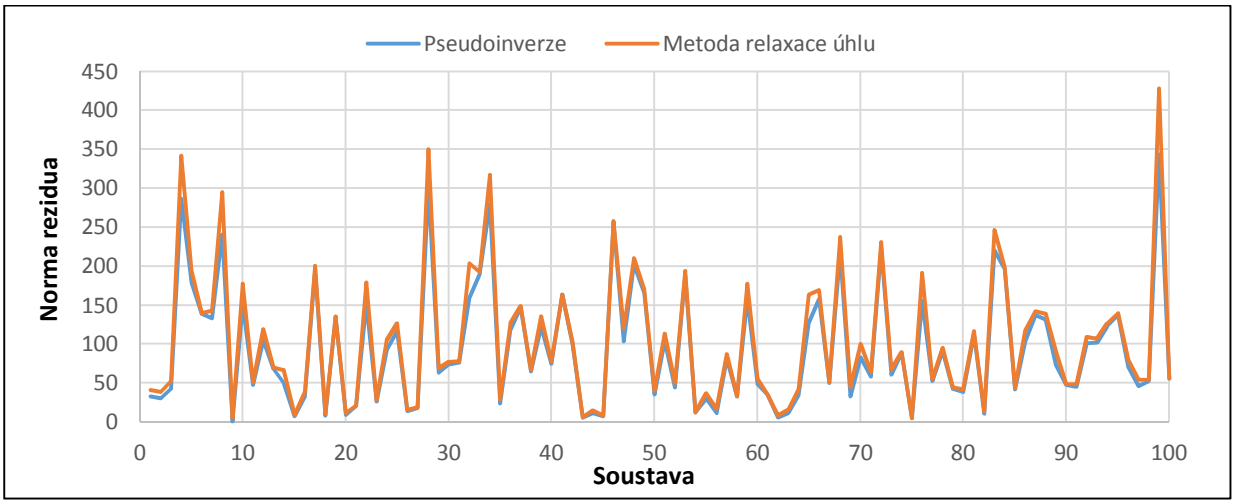
Nabízí se zde otázka, k jakému řešení konverguje metoda relaxace úhlu v případě, kdy má soustava lineárních rovnic nekonečně mnoho řešení $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) < n$, a v případě, kdy žádné řešení nemá $h(\boldsymbol{A}) < h(\boldsymbol{A}|\boldsymbol{b})$. V rámci testu 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. Průběh řešení se postupně ustálí. Ukazuje se, že norma tohoto řešení se blíží k normě, kterou nalezneme pomocí pseudoinverze, tzn. řešení nejmenší v euklidovské normě $\|\boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{x}_{PSD}\|_e \leq \|\boldsymbol{y}\|_e$. 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 10. Na tomto obrázku můžeme vidět, že metoda relaxace úhlu kopíruje co do velikosti $\|\boldsymbol{x}_{RLX}\|_e$ hodnoty, které bychom dostali pomocí pseudoinverze $\|\boldsymbol{x}_{PSD}\|_e$. 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 stejně velkou normu, tzn. $\boldsymbol{x}_{RLX} \neq \boldsymbol{x}_{PSD}$ pro $\|\boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{x}_{PSD}\|_e$. Je to dáno tím, že pracujeme se singulární maticí soustavy, kde každá malá nepřesnost, nebo zaokrouhlení, může způsobit velký rozdíl v řešení soustavy.



Obr. 10: 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. Dosáhnout nulové hodnoty ani nemůže, jelikož žádné přesné řešení soustavy neexistuje. Průběh řešení se postupně ustálí a osciluje kolem určité hodnoty. 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 $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}\|_e$ (obr. 11). Na obrázku 11 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 $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{PSD}\|_e \leq \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{y}\|_e$. Řešení soustavy získané metodou relaxace úhlu opět nemusí být zcela identické s řešením, které získáme pomocí pseudoinverze i pro stejně velkou normu rezidua, tzn. $\boldsymbol{x}_{RLX} \neq \boldsymbol{x}_{PSD}$ pro $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{PSD}\|_e$. Velký vliv má právě oscilace algoritmu. Proto se řešení soustavy mění i při relativním ustálení rezidua kolem minimální hodnoty.



Obr. 11: Norma $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}\|_e$ 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 $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ jedno řešení $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) = n$, tak metoda relaxace úhlu konverguje k tomuto řešení,
- pokud má soustava $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ nekonečně mnoho řešení $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) < n$, tak se metoda relaxace úhlu blíží k řešení nejmenší v euklidovské normě $\|\boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{x}\|_e \leq \|\boldsymbol{y}\|_e$,
- pokud soustava $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ nemá řešení $h(\boldsymbol{A}) < h(\boldsymbol{A}|\boldsymbol{b})$, tak se metoda relaxace úhlu blíží k řešení s nejmenším reziduem v euklidovské normě $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}\|_e \leq \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{y}\|_e$,
- 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. Výhodou je, že v rámci výpočtu inverzní kinematické úlohy v robotice matice nedosahují takových velikostí. Ve většině případech má matice soustavy (Jacobiho matice) maximálně šest řádků a počet sloupců je omezen hodnotou $n \leq 10$, jelikož počet os sériově vyráběných robotů nepřesáhne ve většině případů tuto hodnotu. Více než rychlost je tedy 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 proto často používanou pseudoinverzi nahradit metodou relaxace úhlu a ukážeme si vlastnosti tohoto řešení.

3.1 Inverzní kinematická úloha v robotice

Inverzní kinematická úloha v robotice [9] je zobrazení z prostoru polohy koncového bodu kinematické struktury $\boldsymbol{T}$ do prostoru kloubových souřadnic $\boldsymbol{\theta}$. To znamená, že známe polohu koncového bodu v kartézském souřadnicovém systému $\boldsymbol{T}_R$, resp. homogenní matici transformace $\boldsymbol{T}_H$, a hledáme polohy všech rotačních vazeb $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_n)$ (rovnice 28).

$$\boldsymbol{\theta} = f^{-1}(\boldsymbol{T}) \tag{28}$$

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. 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]. 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.

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 $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$, aby byly naším novým algoritmem relaxace úhlu řešitelné.

Iterační předpis pro Newtonovu metodu s pseudoinverzí Jacobiho matice $\boldsymbol{J}(\boldsymbol{\theta}_k)^+$ je popsán vztahem 29. 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 $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ je popsána vztahem 30.

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \boldsymbol{J}(\boldsymbol{\theta}_k)^+ \cdot \boldsymbol{T}(\boldsymbol{\theta}_k) \tag{29}$$

$$\boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{\theta}_{k+1} = \boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{\theta}_k + \boldsymbol{T}(\boldsymbol{\theta}_k) \tag{30}$$

Metoda pseudoinverze Jacobiho matice vychází z rovnice 31, která popisuje závislost mezi úhlovou rychlostí jednotlivých os a rychlostí koncového bodu v kartézském souřadnicovém systému. Po její linearizaci v nějakém malém okolí koncového bodu kinematické struktury získáme rovnici 32. Změna koncového bodu $\Delta\boldsymbol{T}$ mezi aktuální a žádanou hodnotou je reziduum $\boldsymbol{r}$, které chceme minimalizovat. 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 (33). Často se uvádí v podobě, která odpovídá řešení pomocí metody nejmenších čtverců, tzn. rovnice 33 je ještě vynásobena z obou stran $\boldsymbol{J}^T(\boldsymbol{\theta}_k)$. To pak vede na iterační předpis 34.

$$\frac{d\boldsymbol{T}_R}{dt} = \boldsymbol{J}(\boldsymbol{\theta}) \frac{d\boldsymbol{\theta}}{dt} \tag{31}$$

$$\boldsymbol{J}(\boldsymbol{\theta}) \cdot \Delta\boldsymbol{\theta} = \Delta\boldsymbol{T} = \boldsymbol{r} \tag{32}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{J}(\boldsymbol{\theta}_k)^+ \cdot \boldsymbol{r}_k \tag{33}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{J}^T(\boldsymbol{\theta}_k) \cdot (\boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{J}^T(\boldsymbol{\theta}_k))^+ \cdot \boldsymbol{r}_k \tag{34}$$

Opět můžeme vztah 33 upravit do podoby soustavy lineárních rovnic $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ (35). Je nutné konstatovat, že pro potřeby řešení pomocí metody relaxe úhlu je tento vztah daleko výhodnější, než předpis vycházející z metody nejmenších čtverců (34). 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 $\boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{J}^T(\boldsymbol{\theta}_k)$. Budeme proto preferovat čistou lineární závislost, která z hlediska podmíněnosti matice povede k lepším časovým výsledkům.

$$\begin{aligned} \boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \Delta\boldsymbol{\theta}_k &= \boldsymbol{r}_k \\ \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \Delta\boldsymbol{\theta}_k \end{aligned} \tag{35}$$

Třetí a poslední metodu, kterou upravíme, bude algoritmus Levenberg-Marquardt (36). Nejedná se o nic jiného, než o metodu nejmenších čtverců s tlumením. Iterační předpis 34 je navíc vybaven konstantou λ, která se v robotice používá pro omezení velkých skokových změn v oblasti singularity.

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{J}^T(\boldsymbol{\theta}_k) \cdot (\boldsymbol{J}(\boldsymbol{\theta}_k) \cdot \boldsymbol{J}^T(\boldsymbol{\theta}_k) + \lambda^2 \cdot \boldsymbol{I})^+ \cdot \boldsymbol{r}_k \tag{36}$$

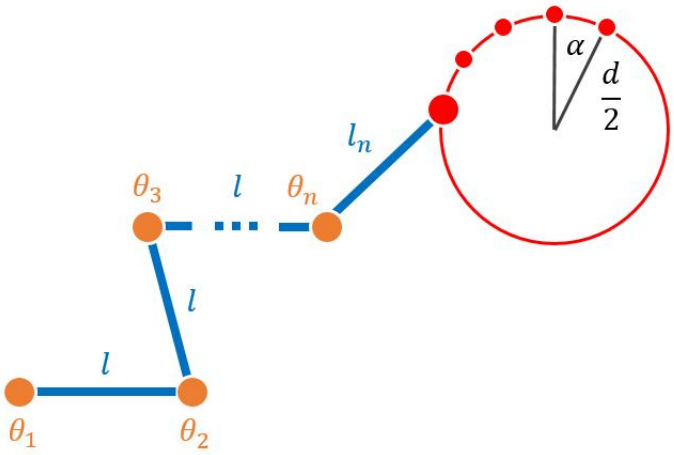
Při úpravě vztahu 36 do podoby soustavy lineárních rovnic se budeme opět snažit obejít zbytečné navýšení čísla podmíněnosti výsledné matice soustavy. Metodu nejmenších čtverců zredukujeme na lineární závislost, ale konstantu tlumení λ ponecháme. Ve výsledku se nejedná o nic jiného, než o rozšíření soustavy o několik dalších rovnic (37). Zapracování konstanty tlumení nás bude stát nějaký iterační čas navíc, jelikož výsledná soustava lineárních rovnic bude o něco větší.

$$\begin{aligned} \begin{pmatrix} \boldsymbol{J}(\boldsymbol{\theta}_k) \\ \lambda^2 \cdot \boldsymbol{I} \end{pmatrix} \cdot \Delta\boldsymbol{\theta}_k &= \begin{pmatrix} \boldsymbol{r}_k \\ \boldsymbol{0} \end{pmatrix} \\ \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \Delta\boldsymbol{\theta}_k \end{aligned} \tag{37}$$

3.2 Porovnání vybraných numerických metod

Teď už nám nezbývá 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 ±0,06 mm. V našem testu převezmeme tuto hodnotu jako podmínku zastavení numerické metody, tzn. $\max|\boldsymbol{r}_k| < 0{,}06$. 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. 12). 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 $\boldsymbol{\theta} = \boldsymbol{o}$, 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 nejčastěji realizovaných robotů. Orientace koncového bodu nebude předepsaná. Trajektorie bude definovaná pouze polohou v rovině x a y.



Obr. 12: Planární manipulátor s n rotačními vazbami a n spoji obkreslující kružnici s úhlovým krokem α

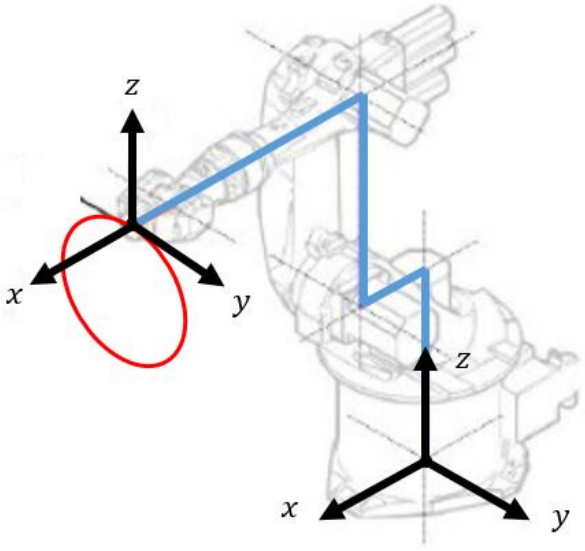
Tab. 5: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice planárního manipulátoru z obrázku 12

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	InverzeJacobiho matice	Levenberg-Marquardt	Newtonova metoda	InverzeJacobiho matice	Levenberg-Marquardt	Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt	Newtonova metoda	Inverze Jacobiho matice	Levenberg-Marquardt
		Ø Čas metody potřebný pro výpočet jednoho úhlového kroku [ms] s požadovanou přesností ±0.06 mm											
π/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
π/50	2	0.58	0.70	0.59	0.57	0.79	0.63	4.67	1.42	1.31	8.74	5.26	3.03
	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
π/100	2	0.56	0.59	0.94	0.55	0.57	0.55	4.52	1.05	1.66	8.56	1.96	2.15
	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 12 jsou uvedeny v tabulce 5. Zde jsou zvýrazněné hodnoty, které nesplnily výše uvedené kritérium řešení do 12 ms. 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. Obě metody generovaly správný průběh žádané hodnoty v čase. Newtonova metoda se ukázala jako nevhodná bez ohledu na postavení manipulátoru. V případě průchodu manipulátoru singulární polohou nejsou časové 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 řídící 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. 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.

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. 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. 6. Rozhodně je zde prostor pro další funkční odladění výpočtu, jelikož metoda relaxace úhlu v maticové podobě zbytečně počítá hodnoty pod a nad diagonálou znaménkové matice.

Z teoretické roviny planárního manipulátoru se nyní pustíme do řešení praktické úlohy s průmyslovým robotem KUKA KR210 R2700 EXTRA. 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. 13). 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 nevychází 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.



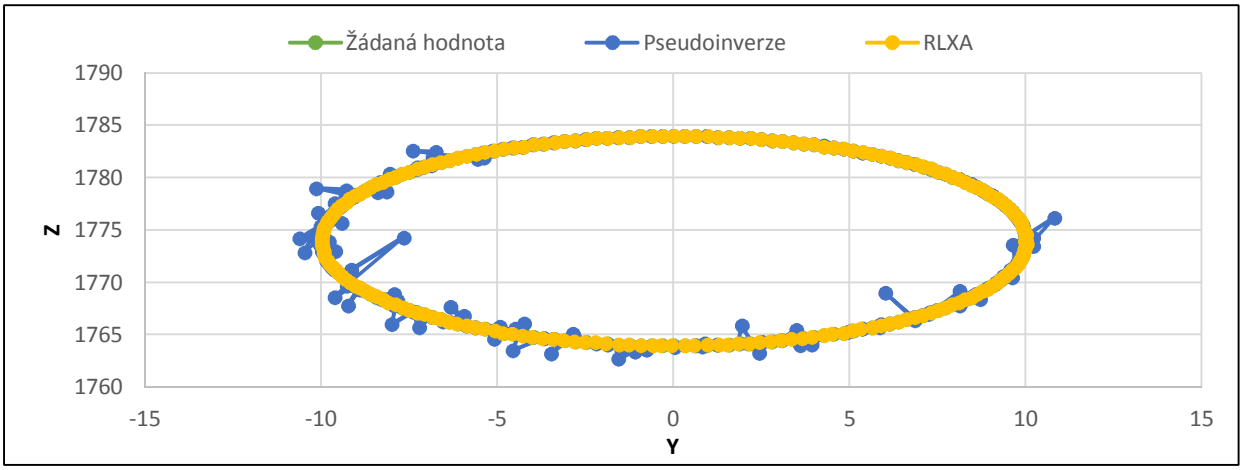
Obr. 13: Robot KUKA KR210 R2700 EXTRA obkreslující kružnici v rovině z a y, obrázek převzat z [29]

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. V oblasti singulární polohy lze pozorovat zpomalení výpočtu u obou metod. 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.

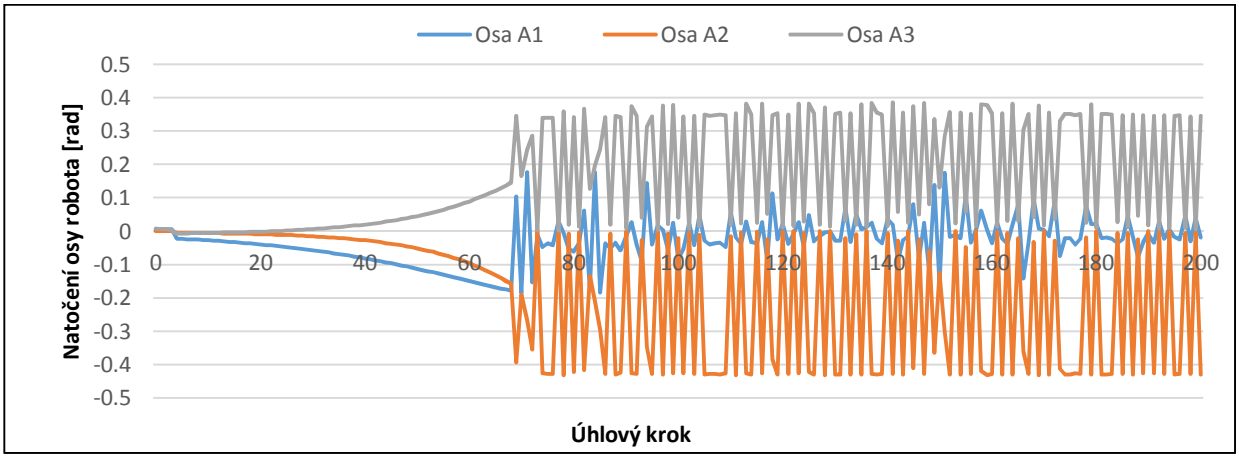
Tab. 6: Časové porovnání vybraných numerických metod při řešení inverzní kinematické úlohy v robotice průmyslového robota KUKA R210 R2700 EXTRA, poloměr kružnice 10 mm

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							
π/25	3.22	2.18	5.06	2.13	4.28	3.36	33.94	52.90
π/50	2.98	1.92	15.52	109.61	3.87	2.71	26.81	40.20
π/100	3.32	1.97	24.77	220.45	3.94	2.55	24.45	28.52

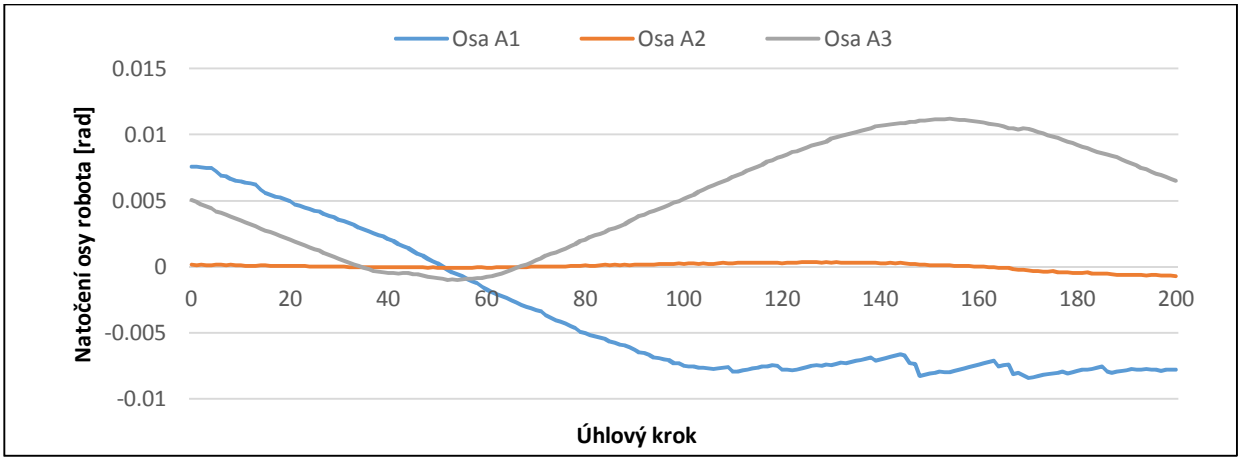
Zajímavé průběhy získáme pro případ počátečního postavení robota v singulární poloze s malým úhlovým krokem výpočtu. Jak už bylo jednou řečeno, pro tuto konfiguraci narůstá čas výpočtu pomocí pseudoinverze a naopak klesá čas výpočtu dosažený metodou relaxace úhlu. Zhoršený čas výpočtu není v tomto případě jediným problémem pseudoinverze. Pseudoinverze začíná špatně sledovat předepsaný průběh žádané hodnoty (viz obr. 14). 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 $\boldsymbol{\theta}_k$ 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. 15, 16). V porovnání s metodou relaxace úhlu je průběh více stochastický. Takový průběh není možné použít jako žádanou hodnotu pro regulační subsystém. Na obrázku 15, 16 jsou uvedeny pro přehlednost pouze průběhy pro první tři osy robota, tzn. pro část kterou nazýváme manipulátor. Pro osy zápěstí je chování obdobné.



Obr. 14: Sledování žádané hodnoty robotem KUKA KR210 R2700 EXTRA, kružnice v rovině y a z s počátkem v singulární poloze s úhlovým krokem π/100, metoda Levenberg-Marquardt



Obr. 15: 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 π/100, metoda Levenberg-Marquardt (Pseudoinverze)

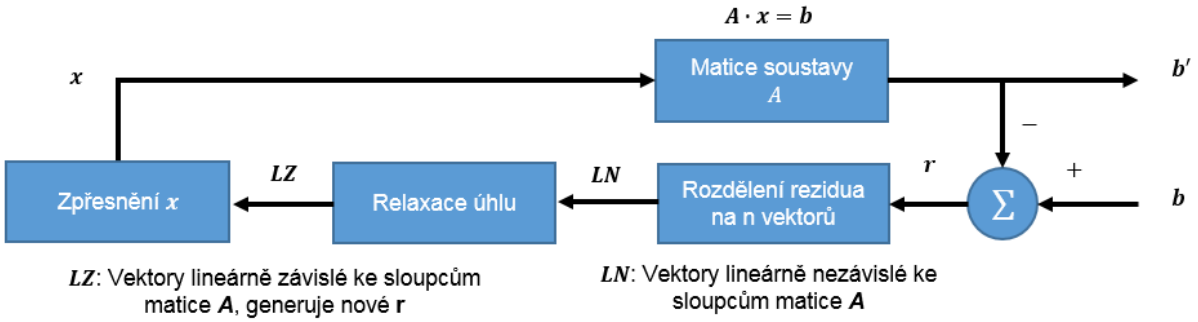


Obr. 16: Průběh osy A1-A3 robota KUKA KR210 R2700 EXTRA při sledování žádané hodnoty, kružnice s počátečním postavením v singulární poloze, poloměr kružnice 10 mm, úhlový krok π/100, metoda Levenberg-Marquardt (RLXA)

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.

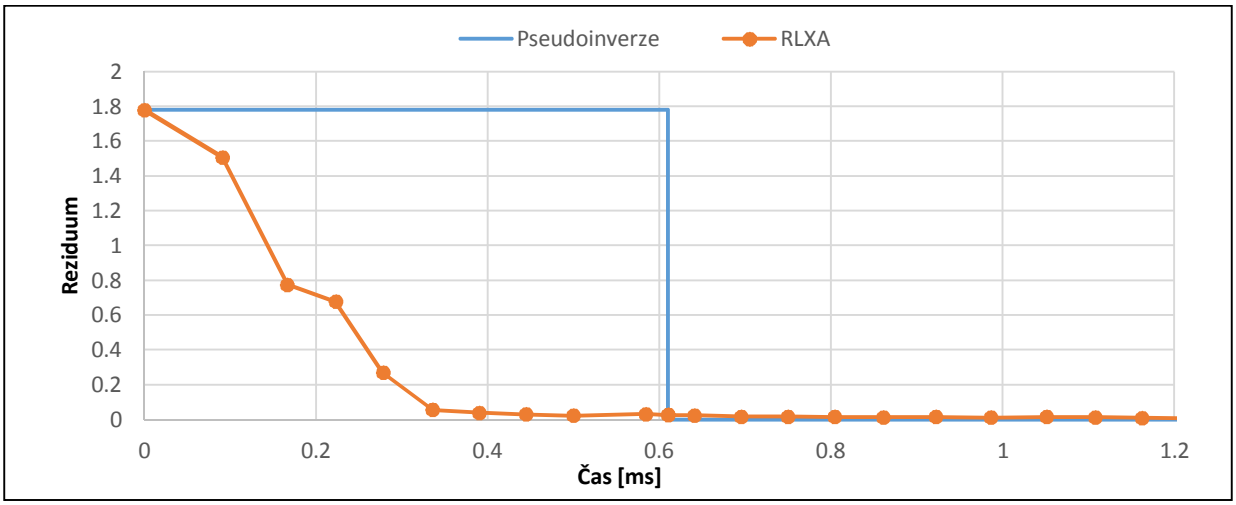
V tuto chvíli musíme odpovědět na otázku, co nám přináší metoda relaxace úhlu v oblasti robotiky oproti pseudoinverzi. Ačkoliv je pseudoinverze velmi dobrou obecnou numerickou metodou, z hlediska řídícího subsystému je daleko zajímavější metoda relaxace úhlu. První důvod jsme již viděli a diskutovali na obr. 15 a 16. 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 $\Delta\boldsymbol{\theta}$. 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 17 je znovu znázorněna základní myšlenka metody relaxace úhlu ve zpětnovazebním zapojení. Pokud se ještě podíváme na zápis algoritmu v maticové podobě $\boldsymbol{P}_{k+1} = \sigma_k \cdot l_k \cdot \boldsymbol{K} \cdot \boldsymbol{Z}_k + \boldsymbol{P}_k$, 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.



Obr. 17: Zpětnovazební zapojení metody relaxace úhlu

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 18, kde je znázorněn časový průběh rezidua $\max|\boldsymbol{r}_k|$ pro obě metody. Z grafu 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. 18: Časový průběh minimalizace rezidua metodou relaxace úhlu a pseudoinverze, matice $n = 2$

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:

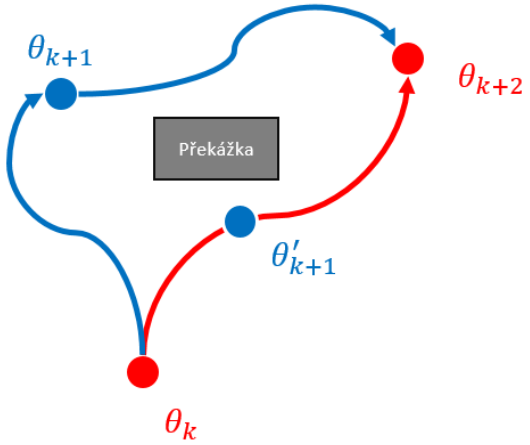
- metodu relaxace úhlu lze použít pro numerické výpočty inverzní kinematické úlohy v robotice v kombinaci s metodou inverze Jacobiho matice a metodou Levenberg-Marquardt. Metodu relaxace úhlu nelze kvůli špatným výsledkům kombinovat s Newtonovou metodou,
- mimo oblast singulárního postavení kinematické 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.

3.3 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.

Pomocí metody relaxace úhly vypočítáme inverzní kinematickou úlohu v robotice a stanovíme natočení jednotlivých os $\boldsymbol{\theta}$. 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 řídící systém robota dostane informaci o nové pozici $\boldsymbol{\theta}_{k+1}$, 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 $\boldsymbol{\theta}_k$ do $\boldsymbol{\theta}_{k+1}$. 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. 19: Průběh ideální (červená) a předepsané (modrá) trajektorie

Trajektorie robota není nikdy popsaná 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. 19. Nyní budeme chtít zjistit, jestli pomocný bod $\boldsymbol{\theta}_{k+1}$ vnáší do trajektorie nějaké časové zpoždění. Podle výše popsaného výpočtu stanovíme nejrychlejší možný čas z $\boldsymbol{\theta}_k$ do $\boldsymbol{\theta}_{k+2}$ (ideální červená trajektorie). Potom stanovíme čas z $\boldsymbol{\theta}_k$ do $\boldsymbol{\theta}_{k+1}$ a z $\boldsymbol{\theta}_{k+1}$ do $\boldsymbol{\theta}_{k+2}$ (předepsaná modrá trajektorie). Pokud bude čas modré trajektorie větší, než čas ideální červené trajektorie, můžeme se pokusit o časovou optimalizaci bodu $\boldsymbol{\theta}_{k+1}$. Tímto způsobem pracuje rutina pro odhalování časových potenciálu v softwaru SKODA TOOL. Obecně neplatí, že přídavný 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.

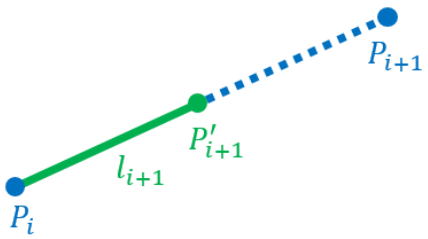
Pokud zjistíme, že bod $\boldsymbol{\theta}_{k+1}$ je možné optimalizovat, pokusíme se nalézt nejrychlejší ideální postavení $\boldsymbol{\theta}'_{k+1}$. Jedná se o bod, který dosáhneme ve stejném čase jako $\boldsymbol{\theta}_{k+1}$, ovšem na červené ideální trajektorii. Pokud neexistuje mezi bodem $\boldsymbol{\theta}_{k+1}$ a $\boldsymbol{\theta}'_{k+1}$ žádná překážka, můžeme bez problémů $\boldsymbol{\theta}'_{k+1}$ prohlásit za nové $\boldsymbol{\theta}_{k+1}$. Získáme tím nejrychlejší možnou trajektorii z bodu $\boldsymbol{\theta}_k$ do $\boldsymbol{\theta}_{k+2}$. V případě, že se mezi bodem $\boldsymbol{\theta}_{k+1}$ a $\boldsymbol{\theta}'_{k+1}$ vyskytuje nějaká překážka, musíme provést pouze částečnou transformaci. Přiblížíme $\boldsymbol{\theta}_{k+1}$ ve směru bodu $\boldsymbol{\theta}'_{k+1}$ 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. Převedeme proto $\boldsymbol{\theta}_{k+1}$ a $\boldsymbol{\theta}'_{k+1}$ pomocí přímé kinematické úlohy v robotice zpět do podoby homogenní matice transformace $\boldsymbol{T}_H$ a $\boldsymbol{T}'_H$, resp. do podoby $\boldsymbol{T}_R$ a $\boldsymbol{T}'_R$. V této podobě můžeme omezit translaci a rotaci koncového bodu robota. Samotná transformace je velice jednoduchá. Určíme vektor $\boldsymbol{v} = \boldsymbol{T}'_R - \boldsymbol{T}_R$, na který aplikujeme omezující podmínky. Tím určíme nový omezený vektor $\boldsymbol{v_o}$. Potom $\boldsymbol{T}'_R = \boldsymbol{T}_R + \boldsymbol{v_o}$. K nalezení bodu $\boldsymbol{T}'_R$ 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.

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. 20). Dále máme definovanou délku l_{i+1}. Celý princip relaxace délky spočívá v přesunutí bodu P_{i+1} ve směru $\boldsymbol{v}_{i+1} = P_{i+1} - P_i$ do vzdálenosti l_{i+1} od bodu P_i tak, že nám vznikne nový bod P'_{i+1}.



Obr. 20: Relaxace délky

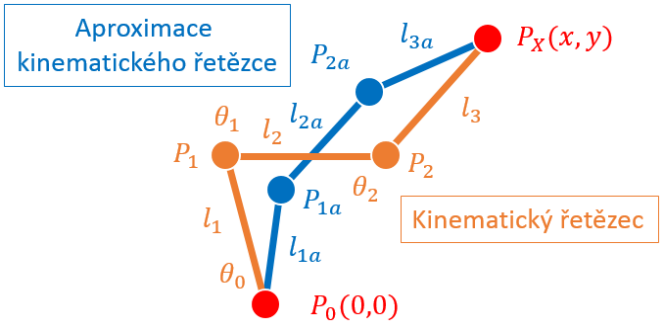
Definice 4.1. Relaxaci délky definujeme jako přesunutí bodu P_{i+1} ve směru $\boldsymbol{v}_{i+1} = P_{i+1} - P_i$ tak, aby metrika $\rho_e(P_i, P'_{i+1}) = l_{i+1}$ (38). Tento jednoduchý výpočet bude základním stavebním kamenem celého algoritmu.

$$P'_{i+1} = \frac{\boldsymbol{v}_{i+1}}{\|\boldsymbol{v}_{i+1}\|_e} l_{i+1} + P_i \tag{38}$$

Můžeme si všimnout, že vztah 38 je velmi podobný vztahu 1 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.

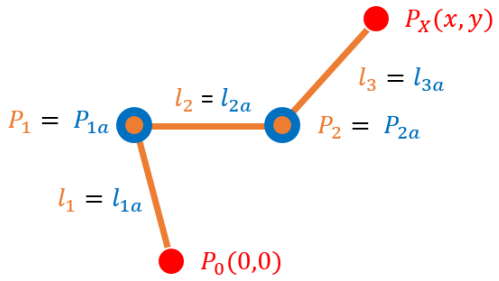
Nyní si vysvětlíme podstatu metody relaxace délky. Uvažujme následující situaci. Máme kinematický řetězec s třemi rotačními vazbami v bodech (P_0, P_1, P_2) a tři kinematické spoje o délce (l_1, l_2, l_3). Tato situace je znázorněna oranžovou barvou na obr. 21. Pozice počátečního bodu P_0 a pozice koncového 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 (θ_0, θ_1, θ_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. 21.



Obr. 21: Aproximační kinematický řetězec

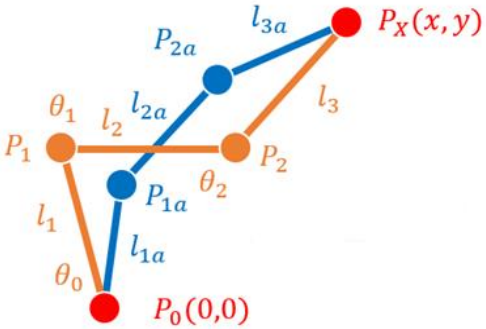
Hlavní myšlenka metody relaxace délky je následující. Pokud nalezneme takové body P_{1a} a P_{2a} aproximačního kinematického řetězce, že $l_1 = l_{1a}$, $l_2 = l_{2a}$ a $l_3 = l_{3a}$, pak tyto body můžeme prohlásit za P_1 a P_2. Celková délka obou řetězců je pak stejná a počáteční bod P_0 a koncový bod P_x je spojen stejnou kinematickou strukturou. Následně můžeme určit úhly všech rotačních kloubů ($\theta_0, \theta_1, \theta_2$), 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. 22.



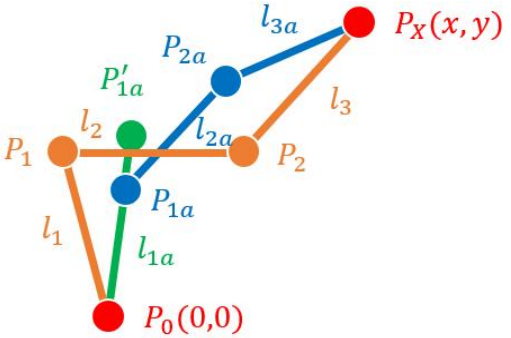
Obr. 22: Optimální řešení metody relaxace délky

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 21 (viz obr. 23).



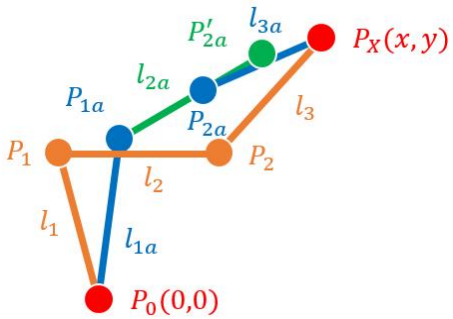
A: Počáteční stav



$$P'_{1a} = \frac{\boldsymbol{v}_{10}}{\|\boldsymbol{v}_{10}\|_e} l_1 + P_0$$

$$l_1 = l_{1a(new)}$$

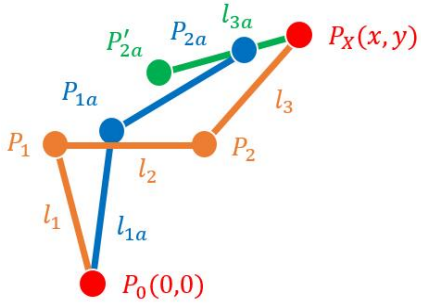
B: Relaxace délky l_{1a} (přímý směr)



$$P'_{2a} = \frac{\boldsymbol{v}_{21}}{\|\boldsymbol{v}_{21}\|_e} l_2 + P_1$$

$$l_2 = l_{2a(new)}$$

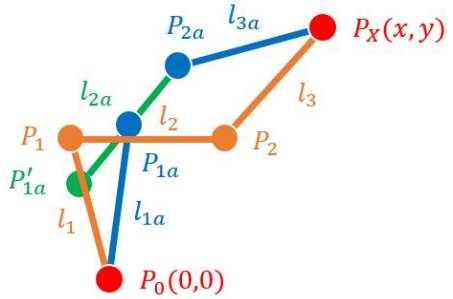
C: Relaxace délky l_{2a} (přímý směr)



$$P'_{2a} = \frac{\boldsymbol{v}_{2x}}{\|\boldsymbol{v}_{2x}\|_e} l_3 + P_x$$

$$l_3 = l_{3a(new)}$$

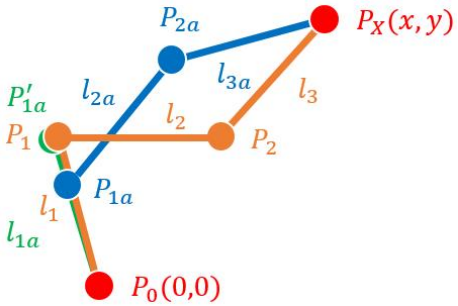
D: Relaxace délky l_{3a} (obrácený směr)



$$P'_{1a} = \frac{\boldsymbol{v}_{12}}{\|\boldsymbol{v}_{12}\|_e} l_2 + P_2$$

$$l_2 = l_{2a(new)}$$

E: Relaxace délky l_{2a} (obrácený směr)



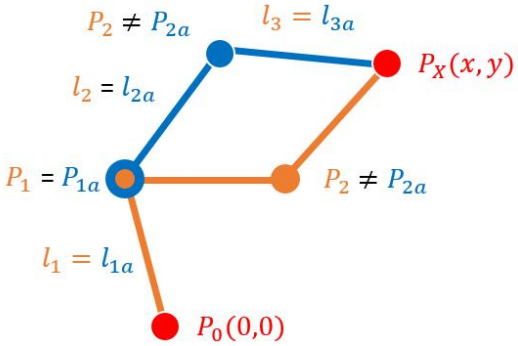
$$P'_{1a} = \frac{\boldsymbol{v}_{10}}{\|\boldsymbol{v}_{10}\|_e} l_1 + P_0$$

$$l_1 = l_{1a(new)}$$

F: Relaxace délky l_{1a} (přímý směr)

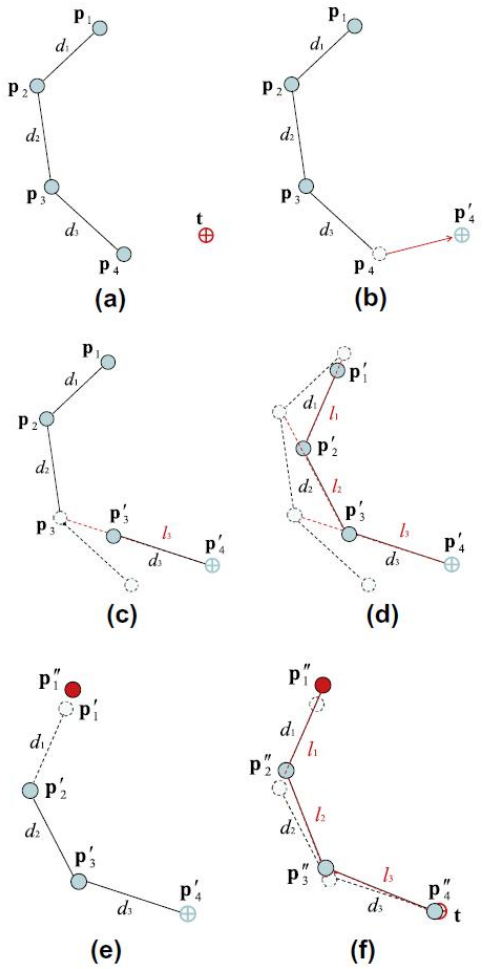
Obr. 23: Postupná relaxace aproximačního kinematického řetězce

Po několika průchodech aproximačním kinematickým řetězcem dosáhneme finální pozice bodů P_{1a} a P_{2a} (viz obr. 24). Celková délka obou kinematických řetězců je sice stejná, ale orientace rotačních vazeb se v tomto případě nerovná. Tento výsledek můžeme nazvat jako neoptimální.



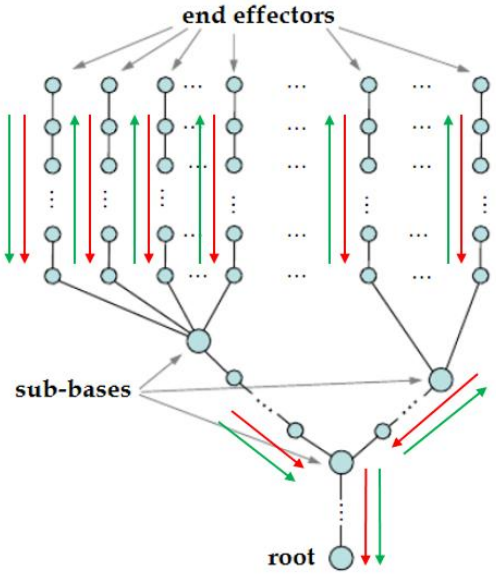
Obr. 24: Výsledek metody relaxace délky (neoptimální)

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. 25 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 25). Metoda relaxace délky nepřesouvá koncové body, čímž šetří výpočetní čas. Proto můžeme hovořit o vylepšené metodě FABRIK.



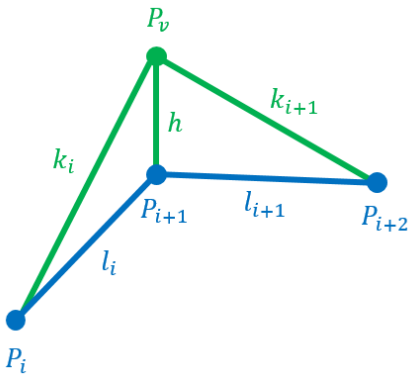
Obr. 25: Výpočet inverzní kinematické úlohy v robotice pomocí metody FABRIK, obrázek převzat z [29]

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. 26. 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 smyslu procházení grafu do šířky. Pokud narazíme na jiný fixní bod, tak procházení dané větvě 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 26 je červenou barvou označen průchod kinematickou strukturou pomocí metody FABRIK a zelenou barvou průchod pomocí metody relaxace délky. Můžeme vidět, jak se obě cesty od sebe liší.



Obr. 26: 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. 27). 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. 27: 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 $\|\boldsymbol{r}\|_e < 0.01$ mm. Počet kinematických vazeb se opět pohyboval v rozmezí $2 \leq n \leq 10$. Výsledné časy jsou uvedeny v tabulce 7. 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. 7: 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 $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ jedno řešení $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) = n$, tak metoda relaxace úhlu konverguje k tomuto řešení,
- pokud má soustava $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ nekonečně mnoho řešení $h(\boldsymbol{A}) = h(\boldsymbol{A}|\boldsymbol{b}) < n$, tak se metoda relaxace úhlu blíží k řešení nejmenší v euklidovské normě $\|\boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{x}\|_e \leq \|\boldsymbol{y}\|_e$,
- pokud soustava $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ nemá řešení $h(\boldsymbol{A}) < h(\boldsymbol{A}|\boldsymbol{b})$, tak se metoda relaxace úhlu blíží k řešení s nejmenším reziduem v euklidovské normě $\|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}_{RLX}\|_e \approx \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{x}\|_e \leq \|\boldsymbol{b} - \boldsymbol{A} \cdot \boldsymbol{y}\|_e$,
- 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 pomoct 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 relace délky pro paralelní kinematické struktury a možnost omezení pohybu kinematické vazby do předepsané roviny. Práce v tomto ohledu představuje novou myšlenku použití přídavné 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 řídící 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 užité 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.yamaha-motor.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]. Europe: 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]. Europe: 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]. Europe: 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