Bakalářská práce
Heuristické metody pro optimalizaci kinematiky robotů/Heuristic methods for robot kinematics optimization

Filip Polák

Akademický rok 2018/2019

PROHLÁŠENÍ

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odoborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne

.....

vlastnoruční podpis

Tímto bych rád poděkoval vedoucímu této bakalářské práce Ing. Martinu Švejdovi, PhD. za jeho vedení, věcné rady, lidskou pomoc a v neposlední řadě nutnou dávku trpělivosti. Dále bych také rád poděkoval své rodině a svým přátelům, kteří mě v tomto období podporovali.

Abstrakt

Tato bakalářská práce se zabývá využitím heuristických metod pro optimalizaci kinematiky robotů. Pro vybraného robota je formulován optimalizační problém pro kritérium optimality - minimalizace sil/silových momentů aktuátorů. Poté je pro robota v Matlabu vyřešen přímý a inverzní kinematický a dynamický problém. Na řešení optimalizačního problému je použitá hrubá síla (prohledávání prostoru pomocí *for* cyklů), metody přímého prohledávání a heuristické metody.

Abstract

The bachelor paper deals with the usage of heuristic methods for robot kinematics optimization. For a particular robot a optimization problem is defined with a certain criteria of optimality - in this case the minimalization of force/moment of force of actuators. Then a direct and an inverse kinematic and dynamic problem is solved in Matlab. To solve the optimization problem one could use brute force (searching the whole are with *for* cycles), direct search methods or heuristic methods.

Obsah

Seznam obrázků

Použité zkratky

DoF (Degrees of Freedom)
Stupně volnosti koncového manipulátoru, jedná se o nezávislé veličiny, které určují stav systému, dají se dělit na translační (posun) a rotační.

SVD (Singular Value Decomposition)
Funkce Matlabu, která rozloží matici na dvě unitární matice a diagonální matici se sestupně seřazenými hodnotami na diagonále.

DGM (Direct Geometric Model)
Přímý kinematický problém (závislost zobecněných souřadnic na kloubových souřadnicích).

IGM (Inverse Geometric Model)
Inverzní kinematický problém (závislost kloubových souřadnic na zobecněných souřadnicích).

IDM (Inverse Dynamic Model)
Inverzní dynamický problém (výpočet síly/silových momentů ze známého pohybu a požadovaných sil na koncový efektor).

DDM (Direct Dynamic Model)
Přímý dynamický problém (výpočet zrychlení na základě polohy, rychlosti a síly/silových momentů působící v kloubech).

QP (Quadratic Programming)
Kvadratické programování, užívá se v případě, kdy má účelová funkce tvar kvadratické funkce

SQP (Sequential Quadratic Programming)
Sekvenční kvadratické programování, užívá se v případě obecně uvažované úlohy, v každém iteraci aproximuje původní úlohu jako úlohu QP.

PSA (Pattern Search Algorithm)
Algoritmus vyhledávání vzoru, metoda přímého prohledávání.

GA (Genetic Algorithm)

Genetický algoritmus, heuristická metoda využívající principů evoluční biologie.

PSO (Particle Swarm Optimization)
Optimalizace roje částic, heuristická metoda hledající kompromis mezi prospěchem jedince a prospěchem roje.

SA (Simulated Annealing)
Simulované žíhání, heuristická metoda využívající analogii k tepelné úpravě materiálu v metalurgii.

RAM Random-Access-Memory
Operační paměť počítače.

Použité matematické značení

$\boldsymbol{X}$ - Zobecněné souřadnice koncového efektoru manipulátoru
Označuje polohu efektoru, kde je poloha dána pozicí (posun v ose x a v ose y) a orientací koncového efektoru (úhel φ). $\dot{\boldsymbol{X}}$ pak označuje rychlost v těchto souřadnicích a $\ddot{\boldsymbol{X}}$ zrychlení v těchto souřadnicích.

$\boldsymbol{Q}$ - Kloubové souřadnice manipulátoru
Vektor souřadnic kloubů manipulátorů $\boldsymbol{Q} = [q_1, q_2, q_3]$, kde q_i označuje natočení i-tého kloubu.

$\boldsymbol{J}$ - Jakobián manipulátoru
$\boldsymbol{J(Q)}$ - jakobián manipulátoru závislý na kloubových souřadnicích manipulátoru.

$\boldsymbol{\tau}$
Vektor sil/silových momentů působící v jednotlivých kloubech.

$\boldsymbol{F}$
Vektor sil/silových momentů působící na koncový efektor.

$\boldsymbol{\lambda}$ Vektor vlastních čísel matice
$\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \lambda_3]$, kde λ_i je i-té vlastní číslo matice.

$\boldsymbol{v}_{\lambda_i}$ - Vlastní vektor
$\boldsymbol{v}_{\lambda_i}$ je vektor odpovídající vlastnímu číslu λ_i a jeho velikost je 1.

$\boldsymbol{I}$ - Matice s jedničkami na diagonále

$\boldsymbol{\sigma}$ - Singulární čísla
$\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \sigma_3]$, kde je σ_i i-té singulární číslo.

$\boldsymbol{U}$ - Unitární matice
$\boldsymbol{U} = [v^L_{\sigma_1}, v^L_{\sigma_2}, v^L_{\sigma_3}]$ - matice, jejíž sloupce tvoří levý singulární vektor.

$\boldsymbol{V}$ - Unitární matice
$\boldsymbol{V} = [v^R_{\sigma_1}, v^R_{\sigma_2}, v^R_{\sigma_3}]$ - matice, jejíž sloupce tvoří pravý singulární vektor.

$\boldsymbol{\Sigma}$ - Diagonální matice
Matice se sestupně uspořádanými singulárními čísly na diagonále.

$\boldsymbol{\xi}$ - Vektor optimalizovaných parametrů
$\boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3] = [L_1, L_2, L_3]$, kde $\xi_i = L_i$ je i-tý optimalizovaný parametr (i-té optimalizované rameno).

$\boldsymbol{\Xi}$ - Přípustná množina návrhových parametrů
Přípustná množina všech ramen.

$\boldsymbol{\tau} = \boldsymbol{IDM}(\boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}, \ddot{\boldsymbol{Q}}, \boldsymbol{\xi}, \boldsymbol{\mu})$ / Řešení IDM
Výpočet sil/silových momentů ze známého pohybu. $\boldsymbol{\xi}$ jsou kinematické návrhové parametry a $\boldsymbol{\mu}$ jsou dynamické návrhové parametry manipulátoru.

$\ddot{\boldsymbol{Q}} = \boldsymbol{DDM}(\boldsymbol{\tau}, \boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}, \boldsymbol{\xi}, \boldsymbol{\mu})$ - Řešení DDM
Výpočet zrychlení kloubových souřadnic na základě známé polohy, rychlosti, sil/silových momentů působící na koncový efektor a sil/momentů sil působocí v kloubech manipulátoru.

$\boldsymbol{\xi_0}$ - Počáteční bod algoritmu
$\boldsymbol{\xi_0} = [\xi_{0_1}, \xi_{0_2}, \xi_{0_3}]$, kde ξ_{0_i} je i-tý počáteční bod.

Kapitola 1

Úvod

Je obecně známé, že se slovem 'robot' se veřejnost seznámila v díle Karla Čapka R.U.R. ('Rossumovi univerzální roboti'), když ho na Karlovu výzvu vymyslel jeho bratr Josef Čapek. Ačkoli by postavy robotů z Čapkovy hry byly z dnešní definice nazývány spíše androidy (roboti, jejichž účelem je vypadat, pracovat a chovat se jako člověk), je až k nevíře, že se s tímto konceptem může člověk setkat již v díle z roku 1920.
Robot je stroj, který pracuje s určitou mírou samostatnosti, vykonává určené úkoly, a to předepsaným způsobem a při různých mírách potřeby interakce s okolním světem a se zadavatelem.
Roboti se dají dělit podle několika kategorií: Podle generace:

- 1. generace - pracují na základě pevného programu
- 2. generace - vybavené senzory a čidly, díky nimž reagují na okolní podmínky

Podle schopnosti se přemisťovat:

- stacionární - nemohou se pohybovat z místa na místo (průmyslové manipulátory)
- mobilní - mohou se přemisťovat (sondy či vozítka na Marsu)

Dále také:

- autonomie (vlastní nezávislost)
- účelu (boj, výroba, průzkum atd.)
- způsob, jakým byly programovány

- podle vzhledu, způsobu vzniku, schopností
  - manipulátor - je ovládán na dálku
  - android - robot podobný člověku
  - droid - jakýkoliv inteligentní a samočinný robot
  - humanoid - robot podobný člověku stavbou těla a hlavně způsobem pohybu
  - anthropomorfní - stroj, který napodobobuje člověka buď fyzicky, způsobem pohybu, nebo i mentálně (počítač HAL 9000 z filmu 2001: Vesmírná odysea)

Jelikož návrh a sestavení robota nejsou triviálním problémem, bude jedním z cílů této bakalářské práce seznámení se základními pojmy, termíny a definicemi, které jsou potřeba znát k pochopení problému známého jako parametrická syntéza. Jedná se o proces, kdy se navrhují jednotlivé části robota – délka ramen, rozměry a typ základny, umístění kloubů atd. Zjednodušeně a ve zkratce – je za úkol navrhnout robota, který bude mít za úkol vykonávat určitou činnost, a na základě zvolení určitých kritérií se takovýto robot navrhne. Je třeba dávat velký pozor na návrh jednotlivých kinematických parametrů, jelikož se může neopatrností a neznalostí dojít k nesmyslnému a nekýženému výsledku. Je důležité brát v potaz i dynamické parametry (hmotnost, momenty setrvačností), které jsou z větší části dány navrženou strukturou, neznamená to ovšem, že není možné někde přidat nebo odebrat část hmotnosti, může se přidat tlumič atd.
Jak již bylo řešeno, nedílnou součástní návrhu a sestavení robota je jeho účel a kritéria, podle kterých tento účel definujeme, čehož lze dosáhnout pomocí *kriteriální funkce*, která v sobě obsahuje účel robota (např. robot má svým manipulátorem dosáhnout na určitá místa svého pracovního prostoru), i kritéria optimality (např. minimalizace sil/silových momentů působící na klouby robota).
Kriteriální funkce je často nespojitá, silně nelineární, nehladká atd., a proto je třeba sofistikovanějších způsobu nalezení jejího minima/maxima, k čemuž slouží *heuristické metody*, které se dokážou s těmito problémy vypořádat a vrací kýžený výsledek.

Kapitola 2

Definice optimalizační úlohy

Jedná se o první a nejdůležitější část procesu optimalizace. V současné době je k ní přistupováno tak, že se zvolí kritérium optimality (minimální silové momenty působící na klouby, minimální projev gravitace působící na ramena atd.), podle něhož se úloha řeší dál. Je důležité si uvědomit, že nesprávným popsáním úlohy lze dojít ve většině případů k nepoužitelným výsledkům, ačkoliv při řešení úlohy neubyla udělána faktická chyba.
Formulace optimalizačního problému pro optimalizaci délek ramen sériového robota se třemi stupni volnosti (DoF - degrees of freedom) pro kompromisní kritérium optimality (kompromis mezi minimalizací sil/ momentů působící na aktuátory a rychlostí kloubů) - též nazývaná kinetostatická dualita - bude následovná:

$$\dot{\boldsymbol{X}} = \boldsymbol{J}(\boldsymbol{Q}) \cdot \dot{\boldsymbol{Q}} \tag{2.1}$$

$$\boldsymbol{F} = \left(\boldsymbol{J}^{\boldsymbol{T}}\right)^{-1} \cdot \boldsymbol{\tau} \tag{2.2}$$

kde $\dot{\boldsymbol{X}}$ je vektor rychlostí zobecných souřadnic (v tomto případě $\boldsymbol{X} = [x, y, \varphi]^T$, kde x je poloha efektoru v horizontálním směru (posun v ose x), y je poloha efektoru ve vertikálním směru (posun v ose y) a φ je natočení efektoru), $\dot{\boldsymbol{Q}}$ je vektor rychlostí kloubových souřadnic ($\boldsymbol{Q} = [q_1, q_2, q_3]$, kde q_1 je natočení prvního kloubu, q_2 je natočení druhého kloubu a q_3 je natočení třetího kloubu), $\boldsymbol{F}$ je vektor sil/silových momentů působící na koncový efektor a $\boldsymbol{\tau}$ je vektor sil/silových momentů působící v jednotlivých kloubech. Jelikož se jedná o 3 DoF neredundantní manipulátor, kinematický jakobián $\boldsymbol{J}(\boldsymbol{Q}) \in \mathbb{R}^{3x3}$ a uvažuje se lineární transformace mezi rychlostmi, což platí pro danou polohu manipulátoru vždy, bez ohledu na redundanci, polohu kloubů atd. Vlastnosti kinematického jakobiánu budou zřejmě dány jeho vlastními čísly λ_1,λ_2 a λ_3 nebo odpovídajícími singulárním číslům σ_1,σ_2 a σ_3. Jelikož jsou sin-

gulární čísla vždy reálná nezáporná čísla, jsou vhodnějším ukazatelem než vlastní čísla.

- Vlastní čísla $\boldsymbol{\lambda} = [\lambda_i]$, vlastní vektory $\boldsymbol{v}_{\lambda_i}$, $\|\boldsymbol{v}_{\lambda_i}\| = 1$:

$$\boldsymbol{\lambda} = \left\{\forall\lambda, \lambda \in \mathbb{C} : det(\lambda \boldsymbol{I} - \boldsymbol{J}(\boldsymbol{Q})) = 0\right\} \tag{2.3}$$

$$\lambda_i \cdot \boldsymbol{v}_{\lambda_i} = \boldsymbol{J}(\boldsymbol{Q}) \cdot \boldsymbol{v}_{\lambda_i}, \quad \forall i \tag{2.4}$$

- Singulární čísla $\boldsymbol{\sigma} = [\sigma_i]$, singulární vektory $v^L_{\sigma_i}$ (levý), $v^R_{\sigma_i}$ (pravý), $\left\|v^*_{\sigma_i}\right\| = 1$:

$$\boldsymbol{\sigma} = \left\{\lambda = \sqrt{\lambda} : det(\lambda \boldsymbol{I} - \boldsymbol{J}(\boldsymbol{Q}) \cdot \boldsymbol{J^T}(\boldsymbol{Q}) = 0)\right\} \tag{2.5}$$

Singulární čísla a vektory lze také definovat pomocí SVD rozkladu jakobiánu:

$$\boldsymbol{J}(\boldsymbol{Q}) = \boldsymbol{U} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T \tag{2.6}$$

kde $\boldsymbol{U} = [v^L_{\sigma_1}, v^L_{\sigma_2}, v^L_{\sigma_3}]$, $\boldsymbol{V} = [v^R_{\sigma_1}, v^R_{\sigma_2}, v^R_{\sigma_3}]$ jsou unitární matice, jejichž sloupce tvoří levý resp. pravý singulární vektor. $\Sigma = diag(\sigma_1, \sigma_2, \sigma_3), \sigma_1 \leq \sigma_2 \leq \sigma_3 \leq 0$ je diagonální matice se sestupně uspořádanými singulárními čísly.

Úpravou vztahu platí:

$$\boldsymbol{U} \cdot \boldsymbol{\Sigma} = \boldsymbol{J}(\boldsymbol{Q}) \cdot \boldsymbol{V} => \boldsymbol{J}(\boldsymbol{Q}) \cdot v^R_{\sigma_i} = \sigma_i \cdot v^L_{\sigma_i}, \quad \forall i \tag{2.7}$$

$$\boldsymbol{V} \cdot \boldsymbol{\Sigma} = \boldsymbol{J}(\boldsymbol{Q})^T \cdot \boldsymbol{U} => \boldsymbol{J}(\boldsymbol{Q})^T \cdot v^L_{\sigma_i} = \sigma_i \cdot v^R_{\sigma_i}, \quad \forall i \tag{2.8}$$

Z teorie indukovaných maticových norem lze ukázat, že platí následující omezení:

$$\max_{\dot{Q}} \left(\frac{\left\| \boldsymbol{J}(\boldsymbol{Q}) \cdot \dot{\boldsymbol{Q}} \right\|}{\left\| \dot{Q} \right\|} \right) = \sigma_{max}(\boldsymbol{J}(\boldsymbol{Q})) = \|\boldsymbol{J}(\boldsymbol{Q})\|_2 = \sigma_1 \tag{2.9}$$

$$\min_{\dot{Q}} \left(\frac{\left\| \boldsymbol{J}(\boldsymbol{Q}) \cdot \dot{\boldsymbol{Q}} \right\|}{\left\| \dot{Q} \right\|} \right) = \sigma_{min}(\boldsymbol{J}(\boldsymbol{Q})) = \sigma_3 \quad \text{(Obecně poslední singulární číslo)} \tag{2.10}$$

kde $\| \star \|$ je Euklidovská norma zobecněných a kloubových rychlostí a $\| \star \|_2$ je norma matice indukovaná Euklidovskou normou.
Platí tedy, že horní a dolní omezení na velikost normy zobecněných rychlostí $\dot{\boldsymbol{X}}$ lze pomocí minimálního a maximálního singulárního čísla jakobiánu psát jako:

$$\sigma_{min}(\boldsymbol{J}(\boldsymbol{Q})) \cdot \left\| \dot{\boldsymbol{Q}} \right\| \leq \left\| \dot{\boldsymbol{X}} \right\| \leq \sigma_{max}(\boldsymbol{J}(\boldsymbol{Q})) \cdot \left\| \dot{\boldsymbol{Q}} \right\| \tag{2.11}$$

Zohlední-li se naopak transformace mezi silami/silovými momenty působícími na koncový efektor manipulátoru a odpovídající síly/silové momenty v kloubech manipulátoru, je tato lineární transformace vztažena transformovaným inverzním jakobiánem $(\boldsymbol{J^T(Q)})^{-1}$. Ze SVD rozkladu lze ukázat (singulární čísla transponované matice se nezmění, zatímco inverze matice hodnoty singulárních čísel převrátí a následně uspořádá v opačném sledu):

$$(\boldsymbol{J^T(Q)})^{-1} = \boldsymbol{U} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T, \quad \boldsymbol{\Sigma} = diag\left(\frac{1}{\sigma_3}, \frac{1}{\sigma_2}, \frac{1}{\sigma_1}\right) \tag{2.12}$$

kde $\boldsymbol{U}$ a $\boldsymbol{V}$ jsou odlišné matice od předchozích a σ_i jsou singulární čísla matice jakobiánu $\boldsymbol{J(Q)}$. Z rovnic (2.2) a (2.11) pak vyplývá:

$$\frac{1}{\sigma_{max}(\boldsymbol{J(Q)})} \cdot \|\boldsymbol{\tau}\| \leq |\boldsymbol{F}\| \leq \frac{1}{\sigma_{min}(\boldsymbol{J(Q)})} \cdot \|\boldsymbol{\tau}\| \tag{2.13}$$

Definice optimalizační úlohy je citována z [1].

2.1 Lokální kritéria

Kritéria vztahující se k jednomu bodu pracovního prostoru. Většina kritérií vychází z vlastností kinematického jakobiánu manipulátoru, ve kterém jsou obsaženy rychlosti kloubových a zobecněných souřadnic a kloubových sil/silových momentů a sil/silových momentů působících na jeho koncový efektor.

2.1.1 Nekonzistentnost fyzikálních jednotek

Vychází z jednoduché skutečnosti – kinematický jakobián v sobě vztahuje vektory s různými fyzikálními jednotkami $[m, \frac{m}{s}, rad, \frac{rad}{s}, N, Nm, ...]$ a číslo podmíněnosti jakobiánu závisí na aktuálně uvažovaných hodnotách. Vyjádříme-li např. translační rychlost v $\frac{mm}{s}$ a úhlovou rychlost v rad, bude se jednat o rozdílné hodnoty a jakobián bude vykazovat nesprávnou podmíněnost. Tomu lze předejít vhodným normováním kinematického jakobiánu, typicky s ohledem na délkové rozměry ramen (normování příslušných translačních rychlostí charakteristickou délkou ramen manipulátoru).
Zřejmým problémem je nalezení vhodných normovacích matic, kterými jsou sloupce/řádky jakobiánu upravovány - proces homogenizace.

2.1.2 Pouze polohové závislosti

Jak je zřejmé už z názvu, kinetostatická kritéria optimality (singulární čísla či číslo podmíněnosti) nebudou zahrnovat dynamické vlastnosti manipulátoru, jelikož jsou

závislá výhradně na jeho aktuální poloze. Kompromisní přístup, kde se jedná o nalezení parametrů manipulátoru s co nejlépe podmíněným kinematickým jakobiánem (včetně metod normování, homogenizace), je vhodný přístup zejména v případě nutnosti obecného vzájemného porovnání zvolených manipulátorů na základě právě kinetostatických ukazatelů.
V reálných úlohách je potřeba dimenzovat manipulátor výhradně pro konkrétní typ úlohy, často pro přesně vymezenou trajektorii pohybu (periodicky se opakující pohyb průmyslových manipulátorů). Zde jsou kinetostatická kritéria nedostačující a je třeba generovat kritéria se zahrnutím kompletních dynamických rovnic chování (závislost nejen na poloze, ale i na na rychlosti a zrychlení kloubových souřadnic).

2.1.3 Maximalizace rychlostí koncového efektoru manipulátoru

Založeno na vztahu (2.11). Norma rychlostí $\left\|\dot{\boldsymbol{X}}\right\|$ koncového eketoru je maximalizována, nebo-li norma rychlostí kloubových souřadnic $\left\|\dot{\boldsymbol{Q}}\right\|$ bude vždy menší než $\frac{1}{\sigma_{min}(\boldsymbol{J(Q)})} \cdot \left\|\dot{\boldsymbol{X}}\right\|$, kde $\left\|\dot{\boldsymbol{X}}\right\|$ představuje požadovanou normu rychlosti koncového efektoru:

$$J(Q) = \sigma_{min}(\boldsymbol{J(Q)}) \rightarrow \text{MAXIMALIZACE} \tag{2.14}$$

2.1.4 Maximalizace sil/silových momentů koncového efektoru manipulátoru

Založeno na vztahu (2.13). Norma síly/silového momentu $\|\boldsymbol{F}\|$ je maximalizována, nebo-li norma sil/silových momentů $\|\tau\|$ působících na kloubech manipulátoru bude vždy menší než $\sigma_{max}(\boldsymbol{J(Q)})\cdot\|\boldsymbol{F}\|$, kde $\|\boldsymbol{F}\|$ představuje požadovanou normu síly/silového momentu koncového efektoru:

$$J(Q) = \sigma_{max}(\boldsymbol{J(Q)}) \rightarrow \text{MINIMALIZACE} \tag{2.15}$$

2.1.5 Podmíněnost kinematického jakobiánu manipulátoru

Jedná se o kompromis, jelikož předešlá dvě kritéria jsou si navzájem protichůdná. V tomto kritériu se snažíme zajistit rovnost minimálního a maximálního čísla jakobiánu. Takovéto kritérium se nazývá podmíněnost jakobiánu a je definováno následovně:

$$\frac{1}{cond(\boldsymbol{J(Q)})} = \frac{\sigma_{min}(\boldsymbol{J(Q)})}{\sigma_{max}(\boldsymbol{J(Q)})} \in \langle 0, 1\rangle \rightarrow \text{MAXIMAIZACE} \tag{2.16}$$

kde cond vrací poměr největšího singulárního čísla matice k tomu nejmenšímu.
V případě $\frac{1}{cond(J(Q)}=1$ se jedná se o izotropní bod manipulátoru, kdy dostáváme ideální přenos sil a rychlostí z aktuátorů manipulátorů na koncový efektor manipulátoru. Pokud $\frac{1}{cond(J(Q))}=0$, je nejmenší singulární hodnota matice rovná 0, a manipulátor se tedy nachází v sériové singularitě, což znamená že se efektor může pohybovat pouze do jediného směru a ve směru kolmém udrží nekonečně velkou sílu/moment síly působící na koncový efektor. $\frac{1}{cond(J(Q))}$ se nazývá dexterity index.

2.1.6 Zahrnutí omezení a vypořádání se se singularitami

Při reálných úlohách dochází často k různým omezením (max/min délka ramen, natočení kloubů atd.), díky čemuž dochází k zkomplikování řešení IGM, která poté může být exaktními přístupy neřešitelná, jelikož mohou mít často s problémovými kriteriálními funkcemi (silně nelineární, nespojité, nehladké, atd.) nebo složitými složitými omezeními mít potíže a nemusí tedy vést k řešení. Tato situace je velmi nebezpečná, jelikož IGM je jedna z prvních řešených úloh a závisí na ní všechny ostatní kinematické a dynamické vlastnosti manipulátoru. Zde také leží největší výhoda užívání heuristických metod, které se zahrnutými omezeními (nejčastěji jsou už započítány do kriteriální funkce - princip penalizace) umí pracovat.

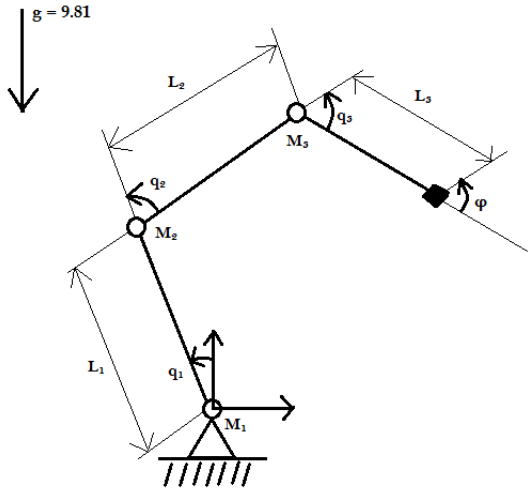
2.2 Globální kritéria

Jak už název napovídá, tato kritéria se řeší po celé trajektorii pohybu nebo na celém pracovním prostoru. Často jsou tato kritéria vypočítávána integrály (přes spojitý prostor) nebo sumací lokálních kritérií (přes diskretizovaný prostor).

Kapitola 3

Vlastní optimalizační úloha

3.1 Popis robota

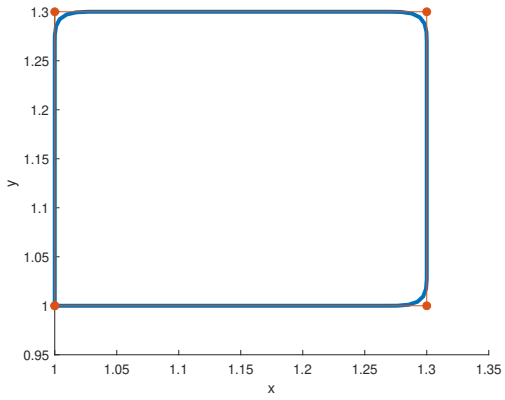


Obrázek 3.1: Model 3DoF robota

Jedná se o 3 DoF neredundatního planárního robota, který má svým manipulátorem (koncem posledního ramena) opisovat předepsanou trajektorii v prostoru. K jeho modelování je zvolen zjednodušený model, kde se jeho ramena popisují jako hmotné tyče s dokonale rovnoměrně rozloženou hustotou a konstantní hmotností a jeho motory (kluby/aktuátory) jako hmotné body s konstatní hmotností.
Ramena robota mají průměr $d = 0.03m$ a hustotu $\rho = 7800\frac{kg}{m^3}$, jedná se tedy o železné tyče. Jejich jednotlivé hmotnosti a pochopitelně tedy i těžiště se dopočítavají v závislosti na jejich aktuální délce L_i.

Jednotlivé otory mají hmotnost nastavenou na $m_1 = 1.5kg$, $m_2 = 1kg$ a $m_3 = 0.5kg$. Na robota působí konstantní gravitační zrychlení $9.81\frac{m}{s^2}$.

3.2 Popis trajektorie



Obrázek 3.2: Generovaná trajektorie

Jak je z grafu zřejmé, trajektorie má tvar čtverce se stranami dlouhými 0.3m a nachází se 1m nad základnou robota. Je ale důležité si uvědomit, že je trajektorie generovaná tak, že ani v jednom rohu kromě počátečního/koncového robot nezastaví, díky čemuž je všech rozích kromě počátečního/koncového trajektorie zaoblená a nedosahuje tedy do samotného rohu.

3.3 Minimalizace sil/silových momentů na kloubech - přes celý prostor

Založeno na vztahu (2.13). Norma síly/silového momentu $\|\boldsymbol{F}\|$ je maximalizována, nebo-li norma sil/silových momentů $\|\boldsymbol{\tau}\|$ působících na kloubech manipulátoru bude vždy menší než $\sigma_{max}(\boldsymbol{J(Q)})\cdot\|\boldsymbol{F}\|$, kde $\|\boldsymbol{F}\|$ představuje požadovanou normu síly/silového momentu koncového efektoru:

$$J(Q) = \sigma_{max}(\boldsymbol{J(Q)}) \rightarrow \text{MINIMALIZACE} \tag{3.1}$$

Nyní je třeba přímo definovat optimalizační úlohu. Je potřeba naleznout minimum

maximální hodnoty kriteriální funkce $J(\boldsymbol{X}, \boldsymbol{\xi})$ podél celého pracovního prostoru pro optimalizaci $\boldsymbol{X} \in \boldsymbol{X}_{opt}$, takže úlohu lze chápat jako maxmin problém:

$$J^* = \min_{\xi \in \Xi} \Big(\max_{\boldsymbol{X} \in \boldsymbol{X}_{opt}} J(\boldsymbol{X}, \boldsymbol{\xi}) \Big) \tag{3.2}$$

$$\xi^* = \operatorname*{argmin}_{\xi \in \Xi} \Big(\max_{\boldsymbol{X} \in \boldsymbol{X}_{opt}} J(\boldsymbol{X}, \boldsymbol{\xi}) \Big) \tag{3.3}$$

kde $\xi = [L_1, L_2, L_3]$ je vektor délek ramen, $\boldsymbol{\Xi}$ je přípustná množina kinematických návrhových parametrů manipulátoru (přípustná množina všech ramen) a J^* je optimální hodnota kriteriální funkce pro nalezenou optimální sadu parametrů ξ^*. Hledáme tedy parametry ξ, pro které platí, že maximální hodnota kriteriální funkce $J(\boldsymbol{X}, \boldsymbol{\xi})$ bude minimalizována podél uvažovaného prostoru $\boldsymbol{X}_{opt}$.

3.4 Řešení přímého a iverzního kinematického a dynamického problému robota

3.4.1 Přímý kinematický problém

Jedná se o nalezení závislosti zobecněných souřadnic $\boldsymbol{X}$ na kloubových souřadnicích $\boldsymbol{Q}$. V literatuře také nazýván direct/forward kinematics problem, direct geometric problem/model (DGM).

3.4.2 Inverzní kinematický problém

Nalezení závislosti kloubových souřadnicích $\boldsymbol{Q}$ na zobecněných souřadnicích $\boldsymbol{X}$. V literatuře inverse kinematic problem, inverse geometric problem/model (IGM).

$$\boldsymbol{Q} = [q_1 \quad q_2 \quad ... \quad q_n]^T$$

kde platí úmluva:

$q_i = \phi_i$ (Kloub je rotačního typu) a $q_i = d_i$ (Kloub je planárního typu)

3.4.3 Inverzní dynamický model (IDM)

$$\left\{\boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}, \ddot{\boldsymbol{Q}}\right\} \rightarrow \boldsymbol{\tau} \tag{3.4}$$

$$\boldsymbol{\tau} = \boldsymbol{IDM}(\boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}, \ddot{\boldsymbol{Q}}, \boldsymbol{\xi}, \boldsymbol{\mu}) \tag{3.5}$$

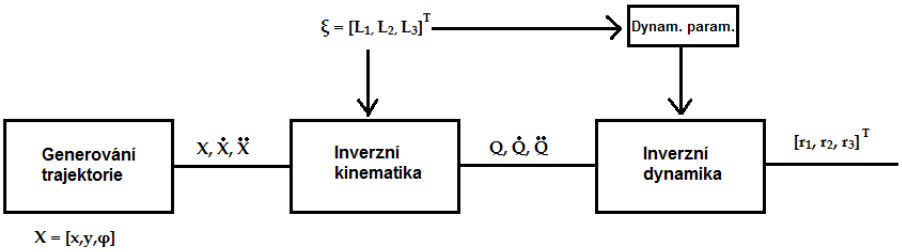
Jedná se o výpočet sil/silových momentů $\boldsymbol{\tau}$ v kloubech manipulátoru ze známého pohybu (polohy $\boldsymbol{Q}$, rychlosti $\dot{\boldsymbol{Q}}$ a zrychlení $\ddot{\boldsymbol{Q}}$) manipulátoru a požadovaných sil/momentů působící na koncový efektor $\boldsymbol{F}$. $\boldsymbol{\tau} = [g_1, g_2, g_3]$, kde g_i je síla/silový moment působící na jeden z kloubů.

3.4.4 Přímý dynamický model (DDM)

$$\left\{\boldsymbol{\tau}, \boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}\right\} \rightarrow \ddot{\boldsymbol{Q}} \tag{3.6}$$

$$\ddot{\boldsymbol{Q}} = \boldsymbol{DDM}(\boldsymbol{\tau}, \boldsymbol{F}, \boldsymbol{Q}, \dot{\boldsymbol{Q}}, \boldsymbol{\xi}, \boldsymbol{\mu}) \tag{3.7}$$

Tedy výpočet zrychlení kloubových souřadnic manipulátoru na základě známé polohy $\boldsymbol{Q}$ a rychlosti $\dot{\boldsymbol{Q}}$, sil/momentů působící na koncový efektor $\boldsymbol{F}$ a sil/momentů púsobící v kloubech $\boldsymbol{\tau}$ manipulátoru. $\boldsymbol{\xi} = [L_1, L_2, L_3]$ jsou kinematické návrhové parametry (délky jednotlivých ramen) a $\boldsymbol{\mu}$ jsou dynamické návrhové parametry manipulátoru. Více informací o těchto problémech lze najít v [1].



Obrázek 3.3: Vizualizace propojenosti jednotlivých výpočtů

Kapitola 4

Řešení optimalizační úlohy

Jedná se obecně o nalezení extrému, ať už lokálního, či globálního, hodnoty kriteriální funkce, která popisuje zvolené kritérium optimality. Samotné řešení zahrnuje řadu metod a způsobů výpočtu (od numerických metod až po umělou inteligenci). Všechny metody užívané k řešení jsou citované z [1].

4.1 Alternativní přístupy k řešení

Jednoduchá myšlenka – původní optimalizační úloha je nahrazena úlohou, k jejímuž řešení lze využít jiných nástrojů a postupů a jejíž řešení lze vhodně aplikovat na úlohu původní.

Kapitola 5

Standardní přístupy k řešení

Jak už bylo řečeno, při řešení optimalizačních úloh využíváme tzv. kriteriální funkci. Ta je obecně definována jako: $J(X, \xi) = J_{obj}(X, \xi) + J_{pen}(X, \xi)$, kde $J_{obj}(X, \xi)$ je účelová funkce založená na zvoleném kritériu optimality a $J_{pen}(X, \xi)$, je penalizační funkce, která v sobě vztahuje penalizační hodnoty odpovídající omezením typu rovností nebo nerovností.

5.1 Lineární programování (LP)

Účelová funkce i omezení jsou zde uvažovány jako lineární vzhledem k optimalizovaným parametrům. Řešení úlohy odpovídá nalezení extrému účelové funkce definované lineární nadplochou na nadrovině definované uvažovanými omezeními.

5.2 Kvadratické programování (QP)

Používá se v případě, že účelová funkce má tvar kvadratické funkce a omezení zůstávají lineární funkcí.

5.3 Sekvenční kvadratické programování (SQP)

V případě obecně uvažované úlohy (nelineární účelová funkce a nelineární omezení nerovnostmi) se využívá SQP, které v každém iteračním kroku aproximuje původní úlohu jako úlohu QP.

5.4 Výhody/nevýhody standardních přístupů

- Jednoduché intuitivní integrování omezení typu rovností i nerovností
- Robustnost algoritmu (stabilní algoritmy optimalizace)
- Možnost vážit omezení (a volně přecházet mezi tvrdými a měkkými omezeními)
- Možnost využití algoritmů optimalizace bez omezení
- Možnost zahrnout v podstatě libovolná omezení

5.5 Gradientní metody

Metody využívané k lokální optimalizaci bez omezení, je ovšem důležité brát v potaz to, že gradientní metody obvykle konvergují do lokálních optim, jejichž globální platnost můžeme dokázat pouze pro silně omezeně definovanou optimalizační úlohu. Pokud chceme najít globální optima, jsou gradientní metody spouštěny z několika různých počátečních podmínek, které můžou být náhodné nebo mohou být ovlivněny určitou heuristikou.

5.5.1 Metoda největšího spádu

Metoda založená na výpočtu gradientu kriteriální funkce a postupu v záporném směru gradientu (= největší spád).

5.5.2 Newtonova metoda

Metoda založená na aproximaci kriteriální funkce funkcí kvadratickou díky Taylorovu rozvoji.

5.5.3 Line search

Metody založené na řešení nedostatků předešlých metod s ohledem na zajištění konvergence algoritmu. V každé iteraci je nalezen směr spádu kriteriální funkce a volí se vhodná vzdálenost posunu tímto směrem.

5.5.4 Trust region

Metody doplňující line search. Nejdříve se nalezne důvěryhodná oblast v prostoru argumentu kriteriální funkce, kde je funkce dostatečně přesně aproximována nějakým modelem. Vzdálenost posunu je pak dána exaktním výpočtem. Např. algoritmus Levenberg – Marquardt: využívá se zde kombinace metody největšího spádu (lepší konvergence, když jsou počáteční podmínky daleko od optima) a Newtonovy metody (lepší konvergence, když jsou počáteční podmínky blízko optima).

5.5.5 Metoda konjugovaných gradientů

Užíváno kvůli tomu, že lze ukázat, že počet potřebných iterací pro výpočet směru kvadratické kriteriální funkce není konečný, jelikož se stává, že v některých následujících iteracích se příspěvek argumentu stornuje, což je způsobeno neortogonalitou směrů vzhledem ke křivosti funkce. Díky metodě konjugovaných gradientů můžeme určit posloupnost směrů a posunů tak, že optimum kvadratické funkce je nalezeno po konečném počtu kroků, který je dán řádem kvadratické funkce. Tato metoda jde zobecnit pro obecné nelineární funkce, jednalo by se ovšem o řešení s obecně nekonečným počtem iterací.

5.5.6 Qasi-Newtonova metoda

Analogie k Newtonově metodě, kde se předpokládá, že nelze efektivně vypočítat Hessián funkce. Může nastat, že aproximaci kriteriální funkce není konvexní funkce, Hessián není tedy pozitivně definitní matice a směr neodpovídá konvergenci k minimu. Princip nalezení aproximace Hessiánu dělí Qasi-Newtonovu metodu na různé varianty: Davidon-Fletcher-Powell, Broyden-Fletcher-Goldfarb-Shanno, Spherical quadratic descent.

5.6 Negradientní metody

Výpočet gradientu (popřípadě Hassiánu) kriteriální funkce může být vzhledem k její složitosti (včetně uvažování penalizací) velmi složité. Můžeme narazit na omezení na použití gradientních funkcí:

- Symbolický výpočet gradientu (Hessiánu) je natolik složitý, že pouze samotné dosazování Hodnot vyžaduje obrovský výpočetní čas a výkon

- Numerický odhad gradientu (Hessiánu) zvyšuje potřebný čas pro iteraci algoritmu – algoritmus se tím stává neřešitelným v rozumném čase
- Kriteriální funkce nemusí být hladka či spojitá – špatná numerická stabilita
- Někdy je obtížná samotná formulace předpisu pro kriteriální funkci (často z hodnot převzaných z experimentu)

5.7 Algoritmy přímého prohledávání (Direct search)

Největší výhoda je to, že stačí znát předpis, který definuje účelovou funkci (nemusíme znát gradient, Hessián).

5.7.1 Pattern search algoritmus (PSA)

Ve své podstatě jednoduchý, ale velmi chytrý algoritmus. Prohledává okolí kriteriální funkce v diskrétních bodech generovaných většinou množinou bázových vektorů definovaných nad definičním oborem kriteriální funkce. V každém kroku se generuje mřížka obsahující tyto body, která vznikne vynásobením směrových vektorů velikostí mřížky a přičtením k současnému argumentu funkce. Poté následuje volba nového argumentu funkce výběrem minimálního bodu z mřížky. V případě, že bod mřížky s minimální hodnotou kriteriální funkce je menší než hodnota aktuálního bodu kriteriální funkce, stává se tím aktuálním bodem (=úspěch), v opačném případě zůstává aktuální bod do další iterace aktuálním (=neúspěch). V případě neúspěchů je možné využít scalování (zvětšování/zmenšování) mřížky. V případě neúspěchu se obvykle velikost mřížky zmenšuje (lepší konvergence), čímž se zmenšuje prohledávaný prostor. Při zvětšení mřížky se rozšiřuje prohledávaný prostor, což napovídá tomu, že je větší šance nalezení globálního optima.

5.7.2 Nelder-Mead algoritmus (NMA)

Založen na heuristickém prohledávání kriteriální funkce. Místo generování mřížky používáme tzv. simplexy (m-rozměrné zobecnění trojúhelníku v definičním oboru kriteriální funkce dimenze m, jedná se o konvexní obal o m + 1 afinně nezávislých bodech). V každé iteraci je generován bod (vrchol) uvnitř nebo vně simplexu, porovná se hodnota kriteriální funkce v tomto bodě s hodnotou kriteriální funkce ve vrcholech simplexu a vrchol s nejhorší hodnotou je vyměněn za generovaný bod. Proces je opakován do chvíle, kdy je velikost simplexu menší než tolerance.

5.7.3 Controlled random search (CRS)

Robustní negradientní algoritmus užívaný pro optimalizační úlohy s obzvláště komplikovanou kriteriální funkcí. Základ tvoří Pure random search (náhodné pokusy na volbu kinematických parametrů), dále je ovšem tento algoritmus obohacen o určité pravděpodobnostní rozdělení a o deterministickou složku, což je změna střední hodnoty uvažovaného pravděpodobnostního rozdělení na základě poslední známé nejlepší hodnoty kriteriální funkce.

5.7.4 Metody typu Monte Carlo

Opět využívány pro složité kriteriální funkce (někdy můžou mít i stochastickou složku) a složitá omezení. Metoda náhodně generuje hodnoty optimalizovaných parametrů a následně vyhodnocuje kriteriální funkci a její omezení.

5.8 Použité metody přímého prohledávání

5.8.1 Pattern search algorithm (PSA)

Byla využita implementece Pattern search algoritmu v Matlabu. Jedním z nejdůležitějších parametrů PSA je tzv. Poll method, která specifikuje vzorec, který algoritmus využívá k vytvoření mesh. Byl použit vzorec GPS Positive Basis 2N, který pro tři proměnné bude vypadat:
$[1\ 0\ 0][0\ 1\ 0][0\ 0\ 1][-1\ 0\ 0][0\ -1\ 0][0\ 0\ -1]$
S tím souvisí Expansion factor, který specifikuje násobek, kterým se mesh zvětší po úspěšném kroku. Expansion factor je nastaven na 2.0. Contraction factor je zase naopak násobek zmenšující mesh po neúspěšném kroku. Je nastaven na 0.5. Při používání Pattern Search Algoritmu je velmi důležitý počáteční bod $\boldsymbol{\xi_0}$. Pokud se nastaví do blízkosti skutečného řešení, je výpočet mnohonásobně rychlejší a přesnější. Po nastavení počátečního bodu na $\boldsymbol{\xi_0} = [0.1\ 1.5\ 3]$ probíhal algoritmus $ellapsed\,time = 201.420s$ a našel nejlepšího hodnotu kriteriální funkce $J(\boldsymbol{\xi}) = 99.8499$ pro optimalizované parametry $\boldsymbol{\xi} = [1.0609\ 0.8789\ 0.0576]$.
Pokud je ovšem počáteční bod nastaven na $\boldsymbol{\xi_0} = [1\,0.9\,0.03]$, což je skutečnému řešení násobně blíže, provede se výpočet za $ellapsed\,time = 140.027s$ a najde nejlepší hodnotu kriteriální funkce $J(\boldsymbol{\xi}) = 99.7838$ pro optimalizované parametry $\boldsymbol{\xi} = [1.0493\ 0.9045\ 0.0347]$.
Pracovní prostor byl omezen dole na $\boldsymbol{lb} = [0.01\ 0.01\ 0.01]$ a nahoře na $\boldsymbol{ub} = [2\ 2\ 2]$

5.9 Heuristické metody

Jedná se o problém, o kterém máme už nějaké povědomí, máme s ním zkušenosti nebo se řídíme jednoduše zdravým rozumem, dá se tedy říci, že používáme tzv. heuristiku. To se může zdát značně zobecněné a neurčité, avšak používání heuristiky se stává nezbytným pro poznání a pochopení toho, že v přírodě jsou komplexní a složité optimalizační úlohy řešeny, ačkoliv jejich model není znám nebo je komplikovaný. Heuristické metody jsou používány hlavně z těchto důvodů:

- Heuristické metody poskytují často uspokojivé výsledky komplikovaných optimalizačních úloh v rozumném čase
- Lze zahrnout omezení
- Vykazují systematické prohledávání stavového prostoru na základě dvoufázového algoritmu – exploration a exploitation
- Kombinace s exaktními metodami – hybridní metody
- Nevýhodou je nutnost vhodné konfigurace těchto metod (ve smyslu nastavení jejich parametrů)

5.9.1 Genetic algoritmus (GA)

Jedny z nejčastějších heuristických algoritmů. Optimalizované parametry (ramena robota) jsou kódovány a generují daného jedince, kterých je na začátku určitá množina a pro každého z nich je vyčíslena hodnota kriteriální funkce. Dle genetických pravidel se z těchto jedinců stávají rodiče, kteří plodí potomky, z nichž se vyberou jedinci s nejlepší hodnotou kriteriální funkce a stávají se z nich rodiče jedinci s nejhorší hodnotou kriteriální funkce zanikají (*stall generations*). Výběr dalších rodičů se řídí některým z pravděpodobnostích přístupů (*roulette wheel selection*, *tournament selection* atd).
Genetická pravidla se typicky rozdělují do dvou tříd. Po *křížení* se proces dostává do fáze prohledávání s velkou množinou působností (*exploration*) a existuje zde řada algoritmů (*one point, multiple point, uniform* atd.). Po *mutaci* proces spadá do fáze algoritmu prohledávání s malou množinou působností (*exploitation*).

5.9.2 Particle swarm optimalization (PSO)

Principiálně podobná metodě GA. Cílem metody je najít kompromis mezi prospěchem jedince (daný hodnotou kriteriální funkce $\boldsymbol{J(\xi)}$ nad vektorem optimalizovaných parametrů $\boldsymbol{\xi}$) a prospěchem celého hejna (daný hodnotami kriteriálních funkcí všech jedinců), což se nazývá *sociální chování*. Na počátku algoritmu jsou generováni jedinci (polohové vektory $\boldsymbol{\xi}_i$) dimenze N, která odpovídá dimenzi optimalizovaných parametrů, a poté se náhodně zvolí jejich současné polohy a rychlosti. Rychlost je kombinací osobního a sociálního chování, to znamená doposud nejlepší hodnotou kriteriální funkce jedince a hejna, což zajišťuje kompromis mezi chováním jedince a celého hejna. Aktualizace polohy $\boldsymbol{\xi}_i$ a rychlosti $\boldsymbol{(v)}_i$ se vyjadřuje:

$$\boldsymbol{\xi}_i^{k+1} = \boldsymbol{\xi}_i^k + \boldsymbol{v}_i^{k+1} \tag{5.1}$$

$$\boldsymbol{v}_i^{k+1} = K\left(\boldsymbol{v}_i^k + \phi_1 \cdot \mathrm{rand} \cdot (\boldsymbol{p_{best}}_i^{k+1} - \boldsymbol{\xi}_i^k) + \phi_2 \cdot \mathrm{rand} \cdot (\boldsymbol{g_{best}}_i^{k+1} - \boldsymbol{\xi}_i^k)\right) \tag{5.2}$$

$$K = \frac{2}{\left\|2 - \phi - \sqrt{\phi^2 - 4\phi}\right\|}, \phi = \phi_1 + \phi_2, \phi > 4 \tag{5.3}$$

kde k je krok algoritmu, K je tlumící faktor, ϕ_1 je koeficient osobního a ϕ_2 je koeficient sociálního chování jedince (logicky tedy část rovnice s ϕ_1 popisuje osobní chování jedince a část rovnice obsahující ϕ_2 popisuje sociální chování jedince), rand je náhodné číslo s rovnoměrným rozdělením, $\boldsymbol{p_{best}}_i^k$ je hodnota polohového vektoru i-tého jedince s nejlepší hodnotou kriteriální funkce v k-tém kroku, $\boldsymbol{g_{best}}_i^{k+1}$ je globální poloha všech jedinců s nejlepší hodnotou kriteriálních funkcí v k-tém kroku algoritmu.

5.9.3 Simulated annealing (SA)

Kombinuje prvky lokálního prohledávání s *Metropolis algorithm* (algoritmus přijímající horší řešení s určitou pravděpodobností). Princip algoritmu je takový, že se generuje kandidát z poslední nejlepší hodnoty, pokud má nový kandidát $\boldsymbol{\xi}_{cand}$ lepší hodnotu kriteriální funkce, nahradí poslední nejlepší hodnotu $\boldsymbol{\xi}_{curr}$. Pokud tomu tak ovšem není, je původní s určitou pravděpodobností nahrazen horším kandidátem. Tato pravděpodobnost p se postupem času zmenšuje, díky čemuž je tato metoda velmi efektivní.

$$p = e^{-\frac{\|J(\boldsymbol{\xi}_{cand}) - J(\boldsymbol{\xi}_{curr})\|}{T}} \tag{5.4}$$

kde proměnná T řízeně klesá v průběhu algoritmu. Je tedy zřejmé, že pro velké T je pravděpodobnost akceptování horšího velká (*exploration*) a s rostoucím časem se zmenšuje (*exploitation*).
Název metody *annealing*, což v češtině znamená žíhání, je analogií k tepelné úpravě materiálu v metalurgii. Rovnice výpočtu pravděpodobnosti p pak odpovídá procesu ochlazovaní materiálu při žíhání.

5.9.4 Gravitational search alogiritmus (GSA)

Vychází z Newtonova zákona gravitačního působení.

$$F = G \cdot \frac{M_1 \cdot M_2}{R^2}, \quad a = \frac{F}{M_1 + M_2} \tag{5.5}$$

kde F je gravitační síla a a je zrychlení, G je gravitační konstanta, M_1 a M_2 jsou hmotnosti těles, která na sebe působí, a R je jejich vzájemná vzdálenost. Poloha i-té částice je reprezentována vektorem optimalizovaných parametrů $\boldsymbol{\xi_i} = [\xi_i^1 ... \xi_i^N]^T$. Princip algoritmu spočívá ve vypočtení gravitační síly působící na každou částici v daném kroku algoritmu a její posun do nové polohy, čemuž odpovídá zrychlení a_i^k za jednotku času. Pro maximalizaci hodnoty kriteriální funkce se dá k-tý krok algoritmu zapsat takto:

- Výpočet hmotnosti M_i i-té částice:

$$m_i^k = \frac{J(\xi_i^k) - J_{worst}^k}{J_{best}^k - J_{worst}^k}, \quad M_i^k = \frac{m_i^k}{\sum_{j=1}^n m_j^k} \tag{5.6}$$

kde $J(\xi_i^k)$ je aktuální hodnota kriteriální funkce, J_{worst}^k resp. J_{best}^k je nejhorší, resp. nejlepší hodnota kriteriální funkce přes celý prostor částic.

- Úprava gravitační konstanty:

$$G^k = G_0 e^{-\alpha \cdot \frac{k}{T}} \tag{5.7}$$

kde G_0 je počáteční hodnota gravitační a T je celkový počet iterací. Je tedy zřejmé, že v průběhu algortimu bude docházet k snižovaní gravitační konstanty v závislosti nejem na T, ale i na konstantě α. Pro velké G_k spadá prohledávání do *exploration* a pro malé G_k do *exploitation*.

- Výpočet působící gravitační síly na částice:
  Pro sílu působící nad d-tou složkou vektoru polohy ξ_i^k i-té částice podle gravitačního zákona platí:

$$F_i^k j[d] = G^k \frac{M_i^k M_j^k}{R_i^2 j + \epsilon}(\xi_j[d] - \xi_i[d]) \tag{5.8}$$

$$F_i^k[d] = \sum_{j \in K_{best}, j \neq i} \text{rand} \cdot F_i^k j[d] \tag{5.9}$$

kde $F_{ij}^k[d]$ je gravitační síla (její d-tá složka) působící na částice ξ_i z částice ξ_j, ϵ je konstanta, $R_{ij} = \|\xi_i - \xi_j\|$ je Euklidovská vzdálenost částic, rand je náhodné číslo v intervalu $< 0, 1 >$, K_{best} je množina prvních K částic s největší hmotností (největší - nejlepší hodnotou kriteriální funkce), které nejvíce ovlivňují vzájemné gravitační působení.

- Úprava polohy i-té částice:
  Zrychlení $a_i^k[d]$, rychlost $v_i^k[d]$ a poloha $x_i^k[d]$ i-té částice jsou dány:

$$a_i^k[d] = \frac{F_i^k[d]}{M_i^k} \tag{5.10}$$

$$v_i^{k+1}[d] = \text{rand} \cdot v_i^k[d] + a_i^k[d] \tag{5.11}$$

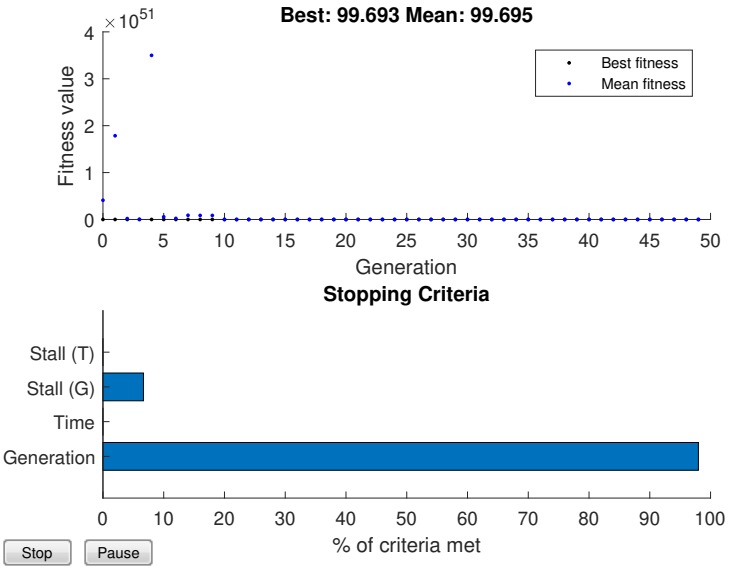
$$\xi^{k+1}[d] = \xi^k[d] + v_i^k[d] \tag{5.12}$$

5.10 Použité heuristické metody

Všechny heuristické metody, které byly použity, jsou již implementovány v Matlabu a výpočty probíhaly v Global Optimization Toolbox, který slouží k hledání globálního minima libovolné funkce. Jelikož ale solvery v tomto prostředí požadují pouze funkce s jedním vstupem, bylo třeba upravit předpis kriteriální funkce *critFcn(ksi,trajectory)*, která přebírala nejenom vektor optimalizovaných parametrů ξ, ale i generovanou trajektorii a čas průběhu trajektorie. Trajektorii a její čas bylo tedy třeba napevno implementovat do kriteriální funkce *critFcn_2(ksi)*, která už přebíra pouze vektor optimalizovaných parametrů. To bylo dosaženo napsáním funkce *create_traj(trajectory)*, která trajetorii, která byla definována jako vektor struktur, převede na čas a na matici cesty, které jsou potom napevno vepsány do kriteriální funkce *critFcn_2(ksi)*. Další důležitou úpravou bylo zachycení neexistence řešení pro takové délky ramen robota, kdy robot nedosáhne na předepsanou trajektorii. Pokud takovýto problém nastal, pak se ve výpočtu inverzní kinematiky objevila odmocnina záporného čísla, kvůli čemuž celý proces skončil. Pokud se ovšem proměnné, která měla být pomocí této odmocniny vypočtena, přiřadí číslo 0, bude tento problém vyřešen, jelikož tyto případy budou vycházet s vysokou hodnotu kriteriální funkce, čímž nebudou zasahovat do hledání globálního minima.

5.10.1 GA

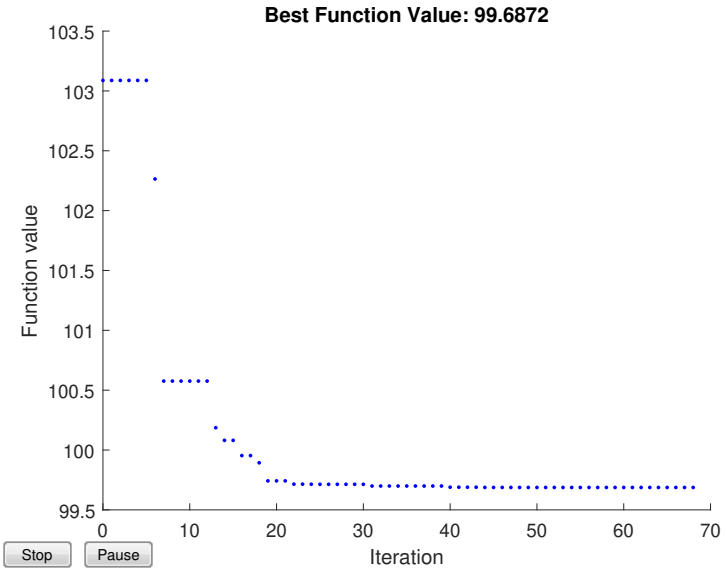
Nejdůležitějším parametrem pro genetický algoritmus je počet generací, který je v Matlabu defaultně nastaven na 50 pro počet proměnných ≤ 5 a na 200 pro počet proměnných > 5. Pro problém 3 DoF planárního neredudantního robota je počet proměnných 3.
Po několika experimentech bylo ukázáno, že se nejlepší hodnota po určité iteraci již nemění, proto byl maximální počet generací nastaven na 50 a maximální počet mrtvých generací na 15. Jelikož byl ošetřen problém neřešitelných délek ramen, není třeba jakkoliv omezovat prostor ramen robota, proto je dolní hranice omezena na $\boldsymbol{lb} = [0.01\ 0.01\ 0.01]$ metru a horní hranice na $\boldsymbol{ub} = [2\ 2\ 2]$ metru.
Simulace probíhala $ellapsed\,time = 856.576s$ a nejlepší hodnota kriteriální funkce byla $J(\boldsymbol{\xi}) = 99.6930$ pro vektor optimalizovaných parametrů $\boldsymbol{\xi} = [0.9965\ 0.9706\ 0.0167]$.



Obrázek 5.1: Genetický algoritmus

5.10.2 PSO

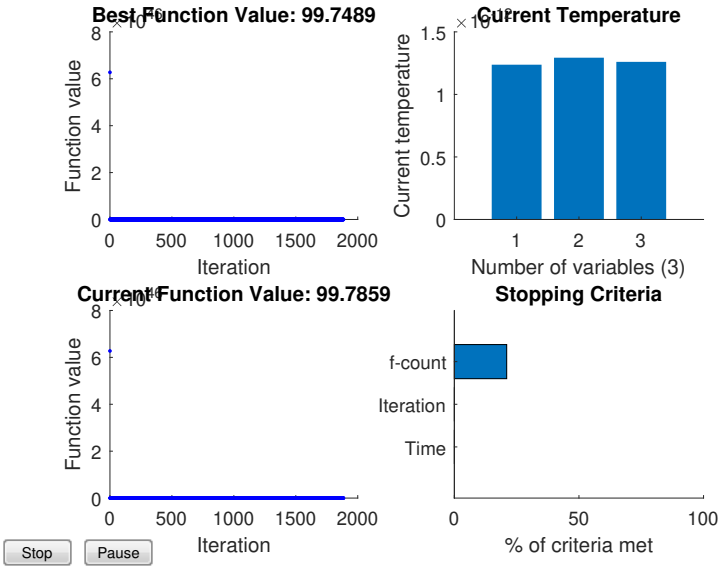
U PSO se nastavuje velikost roje, která je defaultně nastavena na $min(100, 10 * nvars)$, kde nvars je počet proměnných. Velikost roje bude tedy $10 * nvars = 10 * 3 = 30$. Dalším důležitým parametrem je MinNeighborsFraction, který specifikuje sousedství, se kterým algoritmus pracuje, a je nastaven na 0.25.
Stejně jako u genetického algoritmu jsou nastaveny hranice na $\boldsymbol{lb} = [0.01\ 0.01\ 0.01]$ a $\boldsymbol{ub} = [2\ 2\ 2]$.
Proces skončil za $ellapsed\,time = 433.019s$ po 2580 výpočtech funkce, což odpovídá 85 iterícm, a dostal se k nejlepší hodnotě kriteriální funkce $J(\boldsymbol{\xi}) = 99.6872$ pro vektor optimalizovaných parametrů $\boldsymbol{\xi} = [1.0068\ 0.9534\ 0.0281]$.



Obrázek 5.2: Particle swarm optimization

5.10.3 SA

U SA je třeba nastavit počáteční teplotu pro každou proměnnou, defaultně je nastavena na 100 pro každou dimenzi, což se po několika experimentech zdálo jako vyhovující nastavení. Musí se také nastavit teplotní funkce, respektive způsob, jakým se bude ochlazovat. V Matlabu je tato funkce defaultně nastavena tak, že každá další teplota je rovna 0.95 teploty předešlé, čímž se dosáhne pomalého prvnotního ochlazování, postupně bude ovšem docházet k rychlejšímu a rychlejšímu ochlazování. Dalším parametrem algoritmu je tzv. Reanneling, což je část procesu žíhání, který zajišťuje, aby se proces nezastavil na lokálním minimu. Po akceptování určitého počtu nových bodů je teplota procesu zvýšena právě kvůli tomu, aby se algoritmus "nezasekl"na lokálním minimu. Může se ovšem stát, že pokud je Reannealing nastaveno na příliš malou hodnotu, solver nebude schopen najít minimum, proto je vhodnější nastavit jej na vyšší hodnotu. Reannealing je tedy nastaveno na 100.
Pro simulaci žíhání byly použita stejná ohraničení jako u předešlých algoritmů, tedy $\boldsymbol{lb} = [0.01\ 0.01\ 0.01]$ resp. $\boldsymbol{ub} = [2\ 2\ 2]$ pro dolní, resp. horní omezení.
Výpočet trval $ellapsed\,time = 580.520s$ po 1884 iteracích, 1891 výpočtech funkce a nejlepší hodnotu kriteriální funkce $J(\boldsymbol{\xi}) = 99.7489$ pro vektor optimalizovaných parametrů $\boldsymbol{\xi} = [0.9713\ 0.9868\ 0.0323]$.



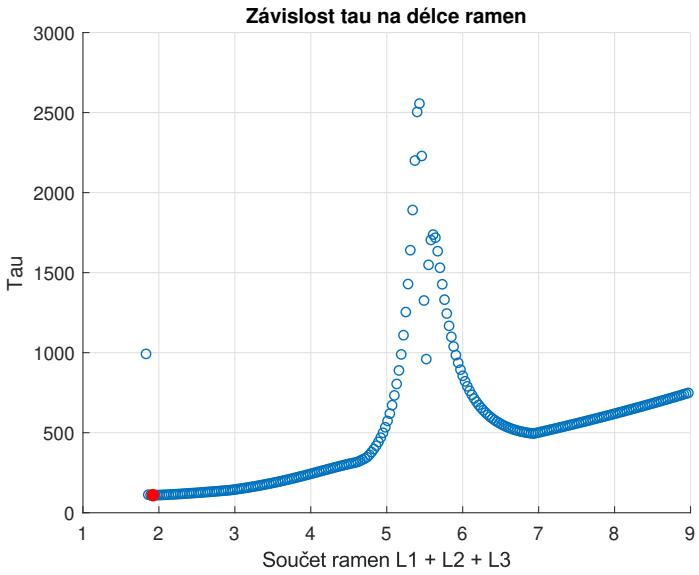
Obrázek 5.3: Simulated annealing

5.11 Řešení hrubou silou

Řešení hrubou silou se rozumí řešení, kdy bude prohledáván pracovní prostor optimalizačních parametrů pomocí *for* cyklů a pro každou kombinaci optimalizačních parametrů (délek ramen) bude vypočtena hodnota kriteriální funkce. Již podle této informace je zřejmé, že tento přístup je velice neefektivní a pomalý.

5.11.1 Ramena mají stejnou délku

V tomto případě lze jednoduše zjistit dolní hranici pracovního prostoru (prohledávaný prostor optimalizačních parametrů). Z výpočtů prováděných v Matlabu vychází, že pokud mají mít ramena stejnou délku, musí každé rameno měřit nejméně $L_i = 0.610208m$, jinak nebudou mít dostatečnou společnou délku na průchod danou trajektorií. Délka ramen byla zvětšována o $0.001m$ a nejlepší hodnota kriteriální funkce $J(\boldsymbol{\xi}) = 110.0381$ byla nalezena pro velikost ramen $\xi_i = L_i = 0.6402m$. Výpočet probíhal $ellapsed\,time = 57.2810s$.



Obrázek 5.4: Graf závislosti normy τ na stejně dlouhá ramena

5.11.2 Ramena mají různou délku

Po několika desítkách výpočtů pomocí heuristických metod je zřejmé, že nejlepší hledané optimalizoavné parametry jsou přibližně $\boldsymbol{\xi} \doteq [1\ 0.9\ 0.03]$, proto byl prohledávaný prostor omezen na $\xi_i \in < 0.02,\ 1.2 >$, aby byla zkrácena doba potřebná k

výpočtu. I přesto je tento výpočet bezkonkurenčně časově nejnáročnější, jelikož při zvětšování ramen o 0.05m, což je v rámci 1m dlouhého ramena nezanedbatelná část, proběhne 13 824 vyhodnocení kriteriální funkce, zápisů do matic atd., což odpovídá $ellapsed\,time = 905.512s$. Nejlepší hodnota kriteriální funkce je $J(\boldsymbol{\xi}) = 99.7702$ pro optimalizované parametry $\boldsymbol{\xi} = [1.020\ 0.9200\ 0.0700]$.

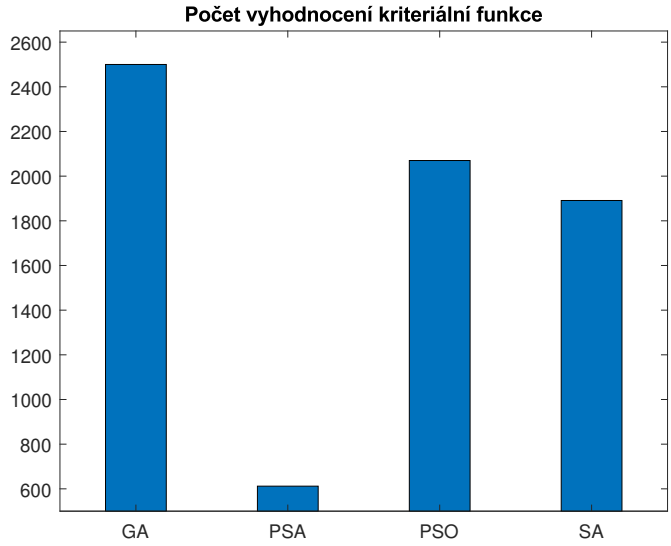
5.12 Porovnání výsledků

Z dostupných výsledků je jisté, že řešení hrubou silou je nejhorší možné řešení, ať už z hlediska přesnosti výsledků, nebo i z časové a výpočetní náročnosti, jelikož především u výpočtů s rameny různé délky dochází k nekýženému zahlcování RAM paměti. Zajímavým poznatkem může ovšem být, že při výpočtech s rameny stejné délky proběhnou výpočty bezkonkurenčně nejrychleji a nejlepší hodnota kriteriální funkce ($J(\boldsymbol{\xi}_{\boldsymbol{ramena\_dohromady}}) = 110.0381$) se liší od té nejlepší hodnoty ($J(\boldsymbol{\xi}) \doteq 99.7$) pouze v rámci jednotek. Při použití algoritmu pro různé délky vyjde mnohem lepší hodnota kriteriální funkce $J(\boldsymbol{\xi}_{\boldsymbol{ramena\_zvlast}}) = 99.7702$, ovšem, jak už řečeno, tento způsob je výpočetně i časově zbytečně náročný.
U přímého prohledávání *Pattern search algorithm* je největší devízou oproti heuristickým metodám její rychlost, která je ovšem velice závislá na vhodném či nevhodném zvolení počátečního bodu algoritmu. Při zvolení počátku, který je blízký skutečnému řešení, je algortimus schopen najít řešení za přibližně 140s, ovšem při nevhodném zvolení počátečního bodu se tento čas může prodloužit. Při vhodném zvolení počátečního bodu je nalezené řešení $J(\boldsymbol{\xi}_{\boldsymbol{psa\_1}}) = 99.7838$, při nevhodném zvolení počátečního bodu pak $J(\boldsymbol{\xi}_{\boldsymbol{psa\_2}}) = 99.8499$.
Genetický algoritmus proběhne za 856.576s a nalezne nejlepší hodnotu kriteriální funkce $J(\boldsymbol{\xi}_{\boldsymbol{ga}}) = 99.6930$.
Particle search optimalizace proběhla za 433.019s a nejlepší hodnotu kriteriální funkce nalezla $J(\boldsymbol{\xi}_{\boldsymbol{pso}}) = 99.6872$.
Výpočet simulovaného žíhání proběhl za 580.520s a nejlepší hodnota kriteriální funkce byla $J(\boldsymbol{\xi}_{\boldsymbol{sa}}) = 99.7489$.

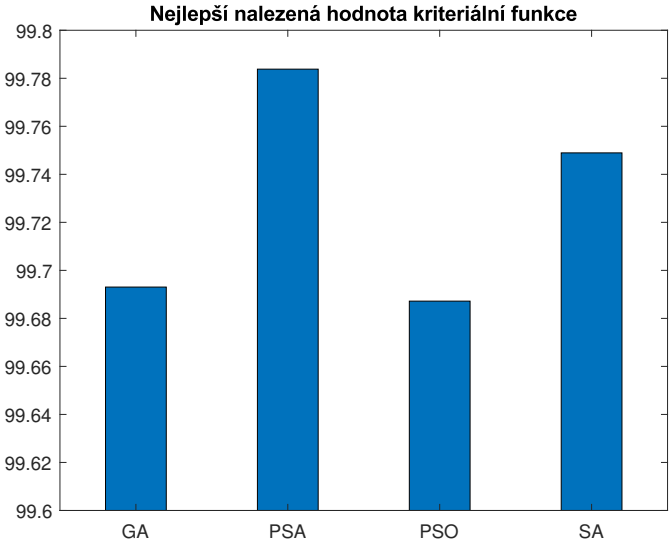
5.13 Shrnutí výsledků

Využívání jednoduchých *for* cyklů je očividně neefektivní z pohledu přesnosti výsledků i času potřebného k výpočtu.
Je ovšem zajímavé, že se při srovnání metody přímého prohledávání a heuristických metod nedojde k prvotně předpokládaným výsledkům. PSA proběhne bezkonkurenčně rychleji, ať už je spuštěn ve vhodném ($\doteq 142s$), či nevhodném počáteční bodu ($\doteq 165s$), a výslednou nejlepší hodnotu nalezne $J(\boldsymbol{\xi}_{\boldsymbol{psa\_1}}) = 99.7838$, což je téměř rovné nejlepší hodnotě kriteriální funkce nalezenou SA ($J(\boldsymbol{\xi}_{\boldsymbol{sa}}) = 99.7489$), ovšem za mnohem delší dobu ($\doteq 580s$). Lepších výsledků než SA dosáhne GA, který je sice pomalejší ($\doteq 856s$),ale nalezne lepší (tzn. nižší) hodnotu kriteriální funkce

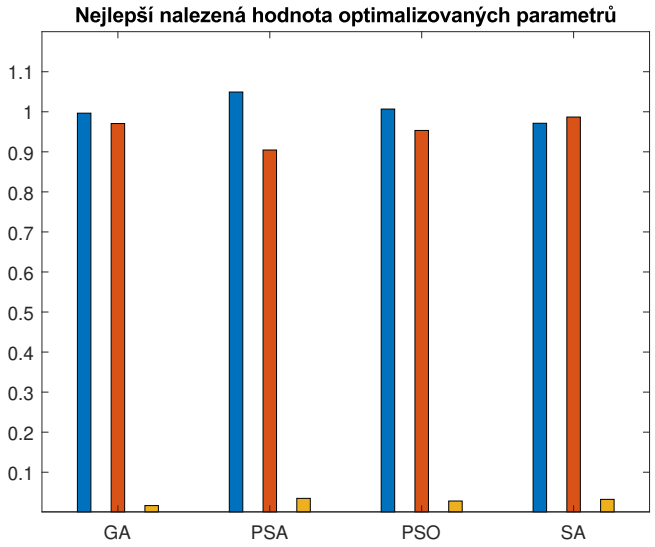
($J(\boldsymbol{\xi_{ga}}) = 99.6930$). Nejlepších výsledků bylo dosaženo pomocí PSO, jež byla jak nejrychlejší ($\doteq 433s$ - "náskok" téměř $1.5m$ před SA), tak nalezne nejlepší výsledek ($J(\boldsymbol{\xi_{pso}}) = 99.6872$).



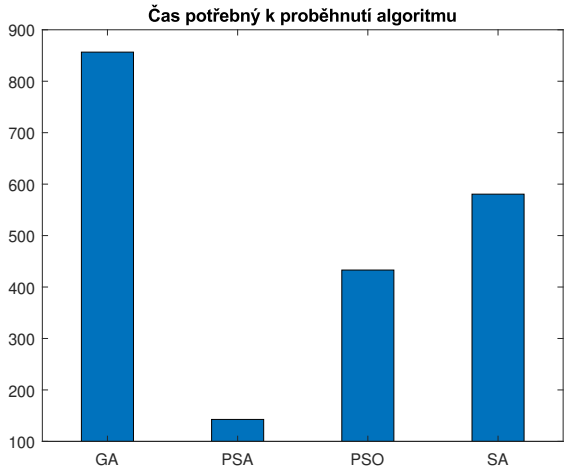
Obrázek 5.5: Počet vyhodnocení kriteriální funkce



Obrázek 5.6: Nejlepší nalezená hodnota kriteriální funkce



Obrázek 5.7: Nejlepší optimalizované parametry



Obrázek 5.8: Čas potřebný k výpočtu algoritmů

5.14 Závěr

Hlavním cílem této bakalářské práce bylo využití heuristických metod pro optimalizaci kinematiky robota. Jednalo se o planárního robota se třemi stupněmi volnosti, který měl za úkol svým manipulátorem (koncem posledního ramena) dosáhnout do všech bodů předem generované trajektorie.

Prvním úkolem byla definice optimalizační úlohy podle určitých kritérií, která je nejdůležitější částí parametrické syntézy, jež zahrnuje všechny kinematické i dynamické informace o robotovi (hmotnost a typ kloubů, typ základny, délka ramen atd.). Jak již bylo řečeno v předešlém odstavci, optimalizační úlohou byl požadavek, aby byl robot schopen svým manipulátorem dosáhnout na všechny body generované trajektorie, a tato úloha se definovala s kritériem minimalizace silových momentů působící na aktuátory (klouby/motory) robota. Tyto požadavky v sobě zahrnuje kriteriální funkce.

V další části bakalářské práce byly v Matlabu vyřešeny přímé a inverzní kinematické a dynamické problémy, díky kterým bylo možné celý systém nasimulovat. V přímém dynamickém problému se hledá závislost zobecněných souřadnic $\boldsymbol{X}$ na kloubových souřadnicích $\boldsymbol{Q}$. V inverzním kinematickém problému je to naopak, hledá se tedy závislost kloubových souřadnic na zobecněných souřadnicích. V inverzním dynamickém modelu se vypočítává silových momentů v kloubech a v přímém dynamickém modelu se vypočítává zrychlení kloubových souřadnic manipulátoru.

Poté může přijít samotné řešení optimalizační úlohy. K tomu je možno přistoupit několika možnými způsoby, kde každý má své výhody a nevýhody. Počítání hrubou silou, standardní metody, gradietní a negradientní metody, metody přímého prohledávání a heuristické metody.

Počítání hrubou silou je velmi jedoduché na implentování, ale je omezené svojí rychlostí. Metody přímého prohledávání mají výhody v tom, že je třeba znát pouze předpis definující účelovou funkci a že jsou rychlé. Heuristické metody jsou vhodné pro řešení komplikovaných problémů, ke kterým dávají výsledky v rozumném čase, lze u nich zahrnout omezení a dají se kombinovat i s exaktními metodami. Nevýhodou je nutnost nastavování a postupné ladění parametrů metod.

Po několika experimentech, úpravách funkcí, ladění parametrů algoritmů a změn ohraničení bylo dosaženo nejlepších výsledků pomocí heuristické metody *Particle search optimization*, která nalezla nejnižší hodnotu kriteriální funkce a z heuristických metod i v nejkratším čase. Heuristická metoda *Genetic algorithm* téměř dosáhla stejného výsledku jako PSO (výsledky se lišily o 0.003), ale z heuristických

metod trvala bezkonkurečně nejdéle. Poslední použitou heuristickou metodou bylo *Simulated annealing*, které bylo sice rychlejší než GA, ale dosáhlo nejhoršího výsledku z heuristických metod. Ačkoli byla metoda přímého prohledávání *Pattern search algorithm* nejrychlejší, její výsledek vykazoval nejvyšší hodnotu kriteriální funkce ze všech použitých metod kromě výpočtu hrubou silou, jenž byl použit hlavně proto, aby se ukázala jeho absolutní neefektivita.

Literatura

[1] Martin Švejda. *Optimalizace robotických architektur*
http://home.zcu.cz/~msvejda/PhD_disertace/SvejdaMartin_thesis_2016_06_14.pdf