
**KAUNO TECHNOLOGIJOS UNIVERSITETAS**
**PANEVĖŽIO TECHNOLOGIJŲ IR VERSLO FAKULTETAS**

**Dovydas Račiūnas**

# Stiuarto platformos judesio modeliavimas

Baigiamasis magistro projektas

**Vadovas**
doc. Lina Urbanavičiūtė

**PANEVĖŽYS, 2015**

Račiūnas, D. Stiuarto platformos judesio modeliavimas. Magistro baigiamasis projektas / vadovas doc. Lina Urbanavičiūtė; Kauno technologijos universitetas, Panevėžio technologijų ir verslo fakultetas, Technologijų katedra.

Panevėžys, 2015. 48 psl.

## SANTRAUKA

Darbe nagrinėjamas lygiagrečiosios kinematinės struktūros manipuliatorius, turintis 6 laisvės laipsnius – Stiuarto platforma, jos modeliavimo ir valdymo galimybės. Analizuojama literatūra, susijusi su šio mechanizmo kinematika, manipuliatorių klasifikacija pagal jo geometrines ir mechanines savybes, dinamika, valdymo sistemomis, modeliavimo galimybėmis, bei pritaikymo sritimis.

Metodologinėje dalyje pateikiami modelio sudarymo etapai, jo geometrinės struktūros išraiška atvirkštinės kinematikos lygtimis, valdymo sistemos, pagrįstos manipuliatoriaus vykdiklių užduočių sudarymu, naudojant atvirkštinį padėčių uždavinį. Nagrinėtosios manipuliatoriaus matematinio modelio lygtys panaudojamos MATLAB ir Simulink programose, sudarant manipuliatoriaus matematinio modelio funkcinių blokų analogą. Apžvelgiami manipuliatoriaus ir jo vykdiklių – hidraulinių servo pavarų mechaninių modelių, naudojant Simulink ir SimMechanics bibliotekas, sudarymo principai. Taip pat pateikiami servo pavaros PID reguliatorių derinimo principai, naudojant kelis populiariausius derinimo metodus.

Tiriamąją dalį sudaro manipuliatoriaus optimalios vykdiklio ilgio paieškos reikiamiems judesiams erdvėje tyrimas. Rastieji optimalūs parametrai naudojami tolesniam atviro kontūro mechanizmo valdymo sistemos tyrimui, siekiant nustatyti jo vykdiklių dinaminius pereinamojo proceso parametrus. Dinaminiai parametrai rasti eksperimentiškai, naudojant liestinės pereinamojo proceso kreivės perlinkio taške ir linijos einančios per du reakcijos kreivės taškus metodus.

**Raktiniai žodžiai**: Stiuarto platforma, lygiagretus manipuliatorius, hidraulinė servo pavara, modelis.

Račiūnas, D. **Modeling of Stewart platform motion**. Masters final project / supervisor doc. Lina Urbanavičiūtė; Kaunas University of Technology, Panevėžys Faculty of Technology and Business, Department of Technologies.

Panevėžys, 2015. 48 psl.

## SUMMARY

In this Master thesis a study of parallel kinematics manipulator with 6 degrees of freedom – Stewart platform, its modelling and control capabilities is made. In this cause its kinematics, classification by its geometrical and mechanical properties, dynamics, control systems and modelling capabilities are being done.

In this study methodological part a stages of mechanism modelling, its geometrical representation with inverse kinematic equations, its joint control system that uses inverse position problem are being proposed. Manipulators mathematical model equations are being used to create their block function analogy in MATLAB and Simulink programs. A review of mechanism and its hydraulic servo actuators mechanical models composition using Simulink‘s and SimMechanics libraries are being done. Also the principles of basic PID tuning using some popular tuning rules overview have been done.

Investigative part of study consists of experimental analysis of optimal actuator length for given required special motions. Received manipulator parameters are being used in open loop mechanism control system test in order to determine its dynamical system properties. Dynamical system reaction parameters are being evaluated by experimental means using tangent line evaluated from system response curve and line going through two pints on system response curve.

**Keywords**: Stewart platform, parallel manipulator, hydraulic servo drive, model.

# TURINYS

# ĮVADAS

Stiuarto platforma (dažnai dar vadinama Stiuarto – Gofo (angl. Stewart-Gough) platforma) yra abstrakcija, naudojama pavadinti specifiniams uždaros mechaninės konstrukcijos, lygiagrečios kinematinės grandinė, robotams. Stiuarto platformos, lyginant su nuoseklios kinematinės grandinės robotais, yra ženkliai mažiau nagrinėtos, bei rečiau pritaikomos. Teorinės šios platformos pritaikymo galimybės yra perspektyvios, dėl to, jog kaip ir įprastiniai nuoseklios kinematinės grandinės robotai, ji taip pat turi šešis laisvės laipsnius, tačiau už įprastus robotus Stiuarto platforma yra pranašesnė savo standumu, svorio ir išvystomos jėgos santykiu, bei tikslumu. Nepaisant to, kaip ir kiekvienas mechanizmas, Stiuarto platforma turi trūkumų – nedidelę darbinę erdvę, bei tam tikroje darbo erdvėje egzistuojančius neapibrėžtumus (kuriuose robotas tampa nevaldomas).

**Tyrimo objektas** – Stiuarto platforma.

**Tyrimo tikslas:** sudaryti Stiuarto platformos valdomo judesio matematinį modelį.

**Tyrimo uždaviniai:**

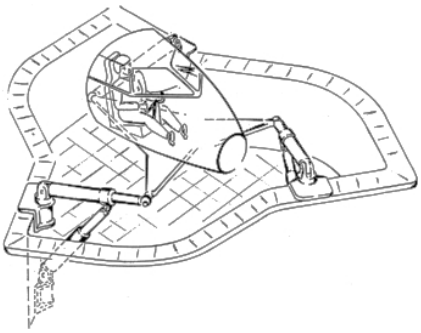
1. Atlikti literatūros, susijusios su Stiuarto platforma, jos modeliavimu ir valdymu, analizę.
2. Sudaryti Stiuarto platformos ir jos hidraulinių servo vykdiklių modelį, naudojantis MATLAB Simulink programiniu paketu.
3. Rasti optimalų vykdiklių ilgį, reikalingą užduotiems judesiams erdvėje atlikti.
4. Išbandyti sudarytą modelį, parinkti ir suderinti vykdiklių judesio reguliatorius.

**Tyrimo metodai** – mokslinės literatūros analizė, modeliavimas kompiuteriu.

# 1. ANALITINĖ DALIS

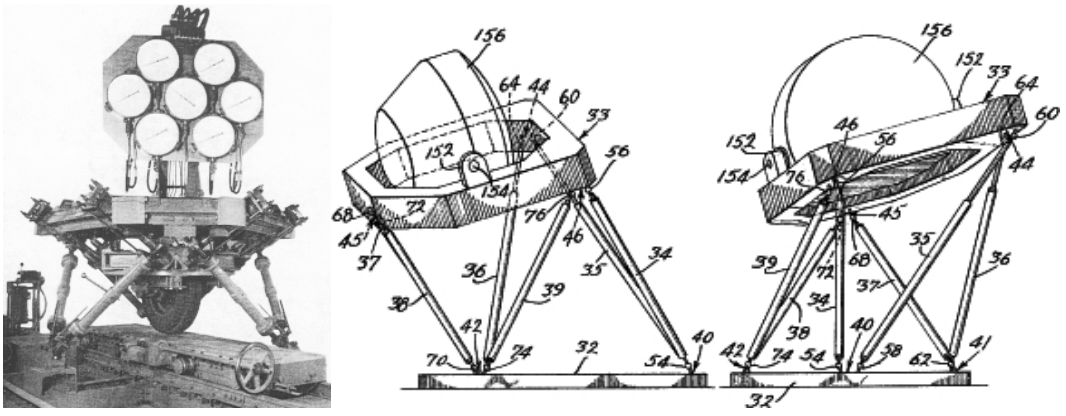
## 1.1 Lygiagrečiųjų manipuliatorių ištakos

Savo darbe D. Stewart [1] pristatė savo sumanytą, šešis laisvės laipsnius turintį manipuliatorių, kuris pagal idėją turėjo būti pritaikytas pilotų mokymui, kaip skrydžio simuliatorius. Stiuarto sumatyto manipuliatoriaus konstrukciją sudarė judanti trikampė platforma ar kabina sujungta su stacionariu pagrindu per tris platformos kojas (gembes), kurių kiekviena yra sudaryta iš dviejų linijinių pavarų (1 pav). Abi linijinės pavaros su manipuliatoriaus pagrindu sujungtos per dviejų ašių sąnaras, dar vadinamas universaliosiomis. Apatinė linijinė pavara su viršutiniąja sujungta per vienos ašies sąnarą, o judanti platforma su viršutiniąja pavara (koja) sujungta per trijų ašių, vadinamąją sferinę, sąnarą. Tokia konstrukcija leidžia kojos sujungimo su judančia platforma tašką valdyti polinėje koordinačių sistemoje, lyginant su manipuliatoriaus pagrindo sujungimu.



1 pav. Stiuarto platformos iliustracija

Pirmieji autoriai, pritaikę šešių kojų konstrukciją lygiagrečios kinematinės konstrukcijos robotams, buvo E. Gough ir K. Cappel. [2]. E. Gough sukurtas manipuliatorius buvo skirtas padangų nusidėvėjimo bandymams (2 pav. a), kaip tik E. Gough manipuliatoriaus konstrukcija




a) b)

2 pav. E. Gough sugalvotas manipuliatorius (a) ir K. Cappel simuliatorius (b)

ilgainiui tapo viena iš plačiausiai paplitusių ir dažniausiai naudojamų konstrukcijų. Kitas pažymėtinas manipuliatorius buvo sukurtas amerikiečių inžinieriaus K. Cappel (2 pav. b), kuris beveik tuo pat metu kaip ir E. Gough, bei D. Stewart sumanė panaudoti šešių kojų lygiagrečios kinematinės grandinės manipuliatorių kaip simuliatorių sraigtasparnių pilotams ruošti. K. Cappel taip pat buvo pirmasis, kuris užpatentavo ši šešių laisvės laipsnių skrydžio simuliatorių.

## 1.2 Manipuliatoriaus mechaninė sudėtis

Stiuarto platformai judesį suteikia šeši vykdikliai jos kojose, nors įmanoma suprojektuoti visą koją, naudojant tik rotacines pavaras, tačiau praktikoje dažniausiai naudojamos linijinės pavaros: pneumatinės, hidraulinės, elektromechaninės, magnetinės, pjezoelektrinės ir pan. Kinematiškai šios pavaros vaizduojamos kaip cilindrinės arba prizmatinės sąnaros, turinčios vieną laivės laipsnį – linijinį poslinkį viena ašimi.

Populiariausios yra elektromechaninės sraigtinės ir hidraulinės linijinės pavaros, tačiau neatsisakoma ir įprastų variklių, servo pavarų [2].

Kinematiškai manipuliatoriaus kojų grandys apibūdinamos jas sudarančiomis sąnaromis ir žymimos sąnarų sutrumpinimais, pavyzdžiui UPS grandį sudaro universalioji, prizmatinė ir sferinė sąnaros. Kojos grandies sudarymui naudojamos tokios sąnaros: rotacinės, prizmatinės, cilindrinės, universaliosios ir sferinės. Nepatariama naudoti SCS ir SPS grandžių, nes jos turi papildomą laisvės laipsnį, dar vadinamąjį pasyvųjį laisvės laipsnį, kuris neįtakoja bendros mechanizmo elgsenos, tačiau yra nepageidaujamas [3].

Tarp lygiagrečių robotų/Stiuarto platformos realizacijų dažniausiai sutinkamos universaliosios, sferinės ir prizmatinės arba cilindrinės sąnaros [3-9].

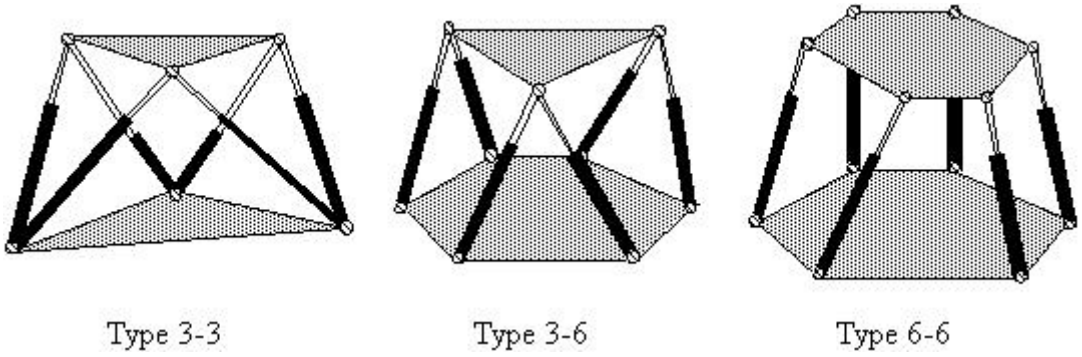
## 1.3 Manipuliatorių klasifikacija

Lygiagrečių manipuliatorių klasifikacijai pagal mechaninę architektūrą nėra sukurta vienareikšmiškos sistemos, todėl jie gali būti apibūdinti įvairiai, tačiau galioja dvi *de facto* sistemos, naudojamos šiems manipuliatoriams klasifikuoti:

1. pagal laisvės laipsnių skaičių ir sąnarų pavadinimų trumpinius;
2. pagal manipuliatoriaus kojų jungimosi su pagrindo ir judančia platforma skaičių.

Tai reiškia, kad tas pats manipuliatorius gali būti apibūdintas dviem būdais, pavyzdžiui, manipuliatorius su šešiomis kojomis, turintis 6 laisvės laipsnius, kurio kojos yra prizmatinės sąnaros, su pagrindo platforma jos jungiamos universaliosiomis sąnaromis trijuose taškuose, o su judančia platforma sferinėmis sąnaromis taip pat trijuose taškuose. Toks manipuliatorius apibūdinamas kaip 6-UPS arba 3-3 manipuliatorius pagal atitinkamas klasifikavimo sistemas. 3-3 konstrukcija dažnai mėgstama pavadinti oktaedru, pagal jos geometrinę formą [2].

Dažniausiai naudojamos geometrinės konfigūracijos yra 3-3, 3-6 ir 6-6, rečiau galima sutikti 6-3 konfigūraciją, galimos ir tarpinės konfigūracijos iš 3, 4, 5 ir 6 jungimosi taškų. Žemiau pateiktame pavyzdyje (3 pav.) matome tris dažniausiai naudojamas manipuliatoriaus konfigūracijas: 3-3, 3-6 ir 6-6.




3 pav. Klasifikavimo pavyzdys

Tačiau šios klasifikavimo sistemos nėra labai informatyvios: pirmuoju atveju 6-UPS apibūdinimas nenurodo geometrinio kojų išsidėstymo, t.y. ar kojos gali dalintis bendru sujungimo tašku; antruoju atveju 3-3 apibūdinimas yra priešingas pirmajam ir nenurodo manipuliatoriaus kojų sąnarų tipo.

## 1.4 Kinematika ir dinamika

### 1.4.1 Atvirkštinis padėčių uždavinys

Norint išvesti atvirkštinės kinematikos lygtį specifiniam manipuliatoriui, pirmiausia manipuliatoriaus pagrindo platformos koordinačių sistema sutapatinama su absoliučiąja koordinačių sistema, po to aprašoma manipuliatoriaus platformų ir kojų sujungimo taškų padėtis koordinačių pradžios taško atžvilgiu. Pagrindo platformos ir judančios platformos taškų koordinatės surašomos į atskiras vektorines matricas $\boldsymbol{b_i}$ ir $\boldsymbol{a_i}$ ($i$=1, ..., 6) atitinkamai. Taip pat reikalinga posūkio matrica $\boldsymbol{R}$, sudaryta iš Oilerio kampų (posūkių) aplink $O_Z$, $O_Y$ ir $O_X$ ašis atitinkamai (1), nurodanti judančios platformos koordinačių sistemos posūkį lyginant su absoliučios koordinačių sistemos pradžios tašku, bei padėties matrica $\boldsymbol{T}$, nurodanti judančios platformos koordinačių sistemos pradžios taško padėtį absoliučioje koordinačių sistemoje (2) [8].

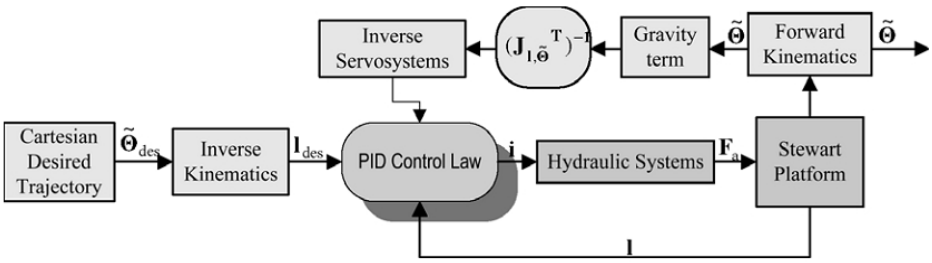
$$\mathbf{R} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma - \sin\alpha\sin\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{bmatrix}. \quad (1)$$

$$\mathbf{T} = \begin{bmatrix} Tx \\ Ty \\ Tz \end{bmatrix}. \quad (2)$$

Tada *i*-tosios platformos kojos vykdiklio ilgis apskaičiuojamas pagal tokią formulę:

$$\mathbf{L}_i = \mathbf{R} \cdot \mathbf{a}_i + \mathbf{T} - \mathbf{b}_i; i = 1,...,6. \quad (3)$$

Sudarius lygčių sistemą kojų ilgiams apskaičiuoti, galima naudoti ją platformos judėjimo trajektorijai generuoti. Iš atvirkštinių kinematikos lygčių gautos vykdiklių padėtys perduodamos reguliatoriui, valdančiam linijinių pavarų padėtis. Dažniausiai tai būna plačiai paplitę PID reguliatoriai [6, 7], tačiau naudojami ir PD reguliatoriai [5] (4 pav.), netiesiniai PID reguliatoriai [8], bei observatoriai [3].




4 pav. Stiuarto platformos valdymo sistemos su PD reguliatoriumi ir gravitacijos kompensavimu modelis (C. Yang. 2010)

### 1.4.2 Tiesioginis padėčių uždavinys

Stiuarto platformai taikomas tiesioginis padėčių uždavinys yra sudėtinga netiesinė lygčių sistema turinti daug realių sprendinių. Tiesioginis padėčių uždavinys lygiagrečiajam manipuliatoriui yra 18 netiesinių lygčių sistema su 6 nežinomaisiais, uždavinio sprendinys – judančios platformos koordinačių sistemos pradžios taškas ir koordinačių sistemos posūkis absoliučios koordinačių sistemos atžvilgiu. Tikslus šių netiesinių lygčių realių sprendinių skaičius nėra žinomas, tačiau moksliniais darbais įrodyta kad tam tikrų konfigūracijų lygiagretieji manipuliatoriai gali turėti nuo 12 iki 40 realių sprendinių [2], dalis realiųjų sprendinių gali būti eliminuota, dėl jų neįmanomos geometrijos. Daugelis darbų, analizuojančių lygiagrečių manipuliatorių tiesioginį padėčių uždavinį, paremti geometrine manipuliatoriaus kinematinės struktūros analize ar supaprastintos struktūros lygčių sprendimu, naudojant ciklinius skaičiavimus, tokius kaip Niutono-Rafsono metodas. Tiesioginis kinematikos uždavinys leidžia iš duotųjų vykdiklių ilgių rasti apibendrintąsias judančios platformos koordinates (tris linijinius

poslinkius ir tris ašinius posūkius absoliučios koordinačių sistemos atžvilgiu). Šios apibendrintosios koordinatės reikalingos valdymo sistemai, skaičiuojant roboto darbinės erdvės ribas ir manipuliatoriaus judėjimo galimybes. Tiesioginio kinematinio uždavinio lygtis gali būti išvedama iš (3) lygties, ieškant nežinomų poslinkio ***T*** ir posūkio ***R*** matricų**.** Įprastai spręsti tiesioginės kinematikos lygtims taikomas Niutono-Rafsono metodas [5]. Ieškant tiesioginės kinematikos lygčių sprendinių, šiuo metodu naudojama (4) formulė.

$$\mathbf{q}_{j+1} = \mathbf{q}_j + \mathbf{J}_{l\Theta}^{-1} \cdot (l_0 - l_j); \tag{4}$$

čia: ***q*** – manipuliatoriaus judančios platformos apibendrintosios koordinatės, ***J*** – Jakobiano (dalinių išvestinių) matrica, $l_0$ – pradinis vykdiklio ilgio spėjimas, $l_j$ – vykdiklio ilgio spėjimas *j*-tosios iteracijos metu.

C. Yang (2009) savo darbe [4] aprašė Stiuarto platformos valdymo sistemą, naudojančią visuotinius Niutono-Rafsono algoritmus su monotoniniu nuolydžiu – GNRMD (*angl. Global Newton – Raphson with Monotonic Descent Algorithms*), skirtą tiesioginės kinematikos uždavinio netiesinių lygčių sprendiniui rasti realiu laiku. Pasak autoriaus, GNRMD algoritmai pašalina Niutono – Rafsono algoritmų trūkumą – algoritmas apskaičiuoja dalinį sprendinį, esantį arčiausiai pradinės aproksimacijos. GNRMD algoritmas klasikinį Niutono-Rafsono algoritmą papildo monotoninio nuolydžio operatoriumi *w*, kuris algoritmui yra:

$$\mathbf{q}_{j+1} = \mathbf{q}_j + w_j \cdot \mathbf{J}_{l\Theta}^{-1} \int_{\Theta_j} (l_0 - l_j), j = 0,1,2,\ldots; \tag{5}$$

čia $w_j \in R$ - monotoninio nuolydžio operatorius kiekvienoje iteracijoje, $0<w\leq 1$.

Valdymo sistema su GNRMD algoritmu buvo pritaikyta realaus laiko pramoniniam kompiuteriui, programa parašyta MATLAB programiniu paketu ir įvykdyta MATLAB RTW/xPC operacinėje sistemoje. Algoritmas apribotas tokiomis sąlygomis:

$$\begin{cases} max\left|\{h(\mathbf{q}_j)\}\right| < \varepsilon \\ j \leq N \end{cases}; \tag{6}$$

čia $\varepsilon$ – ribinė tolerancija, $N$ – didžiausias iteracijų skaičius.

Šios sąlygos (5), (6) užtikrina valdymo sistemos veikimą realiu laiku ir apsaugo nuo begalinės skaičiavimų iteracijos.

### 1.4.3 Kinematiniai greičių uždaviniai ir Jakobianas

Bendruoju atveju robotų kinematinių greičių uždavinių teorijoje Jakobianas yra dalinių išvestinių matrica, susiejanti apibendrintųjų grandžių greičius jų koordinačių sistemose su judančios platformos greičiu absoliučioje koordinačių sistemoje. Atvirkštinis Jakobianas randamas pirmiausia, diferencijuojant atvirkštinio kinematinio padėties uždavinio (3) lygtį laiko atžvilgiu ir taip surišant vykdiklių greičius $q'$ su platformos greičiu $X'$:

$$\frac{\delta \dot{\mathbf{q}}}{\delta t} = \mathbf{J} \cdot \frac{\delta \mathbf{X}}{\delta t} \rightarrow \dot{q} = \mathbf{J} \cdot \dot{\mathbf{X}}; \tag{7}$$

čia: $\boldsymbol{X} = [x,y,z,\psi,\theta,\varphi]^{\mathrm{T}}$ – apibendrintas padėties vektorius absoliučioje koordinačių sistemoje, $\boldsymbol{X'} = [v,\omega]^{\mathrm{T}}$ – apibendrintas greičių vektorius.

Išreiškus kojos vektorių, jo ilgio skaliaro $\boldsymbol{q}_i$ ir krypties vektoriaus $\boldsymbol{e}_i$ sandauga, ir (3) formulės išvedama lygtis reikalinga Jakobiano apskaičiavimui:

$$\dot{\mathbf{q}}_i = \mathbf{e}_i \cdot \mathbf{v} + (\mathbf{R} \cdot \mathbf{a}_i \times \mathbf{e}_i) \cdot \boldsymbol{\omega}. \tag{8}$$

Iš šios lygties sudaroma 6x6 Jakobiano matrica:

$$\mathbf{J} = \begin{bmatrix} \mathbf{e}_1^T & (\mathbf{R} \cdot \mathbf{a}_1 \times \mathbf{e}_1)^T \\ \dots & \dots \\ \mathbf{e}_6^T & (\mathbf{R} \cdot \mathbf{a}_6 \times \mathbf{e}_6)^T \end{bmatrix}. \tag{9}$$

Gautoji Jakobiano matrica leidžia apskaičiuoti judančios platformos padėties vektoriaus kitimo greitį, žinant vykdiklių padėčių kitimo greičius. Kinematiniai greičių uždaviniai yra išsprendžiami tik tada, kai Jakobianas yra 6x6 dydžio matrica, todėl šis Stiuarto platformos parametras yra naudojamas dar vienam svarbiam tikslui – rasti manipuliatoriaus trajektorijos sritis, kuriose jis tampa nevaldomas. Jakobianas taip pat gali būti išskirtas į dvi dedamąsias:

$$\mathbf{J} = \mathbf{J}_q^{-1} \cdot \mathbf{J}_X. \tag{10}$$

tada (7) lygybė gali būti išreikšta:

$$\mathbf{J}_q \cdot \dot{\mathbf{q}} = \mathbf{J}_X \cdot \dot{\mathbf{X}}. \tag{11}$$

Manipuliatoriaus nevaldomumo nustatymas atliekamas, skaičiuojant Jakobiano dedamųjų determinantus, todėl yra dvi situacijos, kada robotas gali tapti nevaldomas:

- jei $\boldsymbol{J}_q$ dedamosios determinantas yra lygus 0, mechanizmas praranda vieną laisvės laipsnį ir tampa nevaldomas.
- jei $\boldsymbol{J}_x$ dedamosios determinantas yra lygus 0, mechanizmas įgauna vieną papildomą laisvės laipsnį ir taip pat tampa nevaldomas.

Pažymėtina, jog šešiakojams lygiagretiems manipuliatoriams $\boldsymbol{J}_q$ Jakobianas yra vienetinė matrica, todėl šio tipo manipuliatoriams galimas tik vieno tipo nevaldomumo atsiradimas.

### 1.4.4 Dinamika

Manipuliatoriaus dinamikos lygtys yra reikalingos norint sudaryti manipuliatoriaus matematinį modelį, kuris panaudojamas valdymo sistemoje, įvertinant veikiančias jėgas ir manipuliatoriaus judesius, taip pat dinaminiam roboto judesių modeliavimui.

Dinamikos lygtys ir modelis gali būti išvestos naudojantis Lagranžo, Niutono-Oilerio ir Keino metodais [5], [9].

Lagranžo metodas remiasi roboto dekompozicija į judančią platformą ir roboto kojas, šioms dviem dekomponuotoms sistemoms apskaičiuojamos kinetinės ir potencinės energijos (linijiniam ir rotaciniam judesiui) ir iš jų išvedamos manipuliatoriaus dinamikos lygtys [10]. Lagranžo formuluotė apibendrintoms koordinatėms atrodo taip:

$$\frac{\mathrm{d}}{\mathrm{d}t}\cdot\frac{\mathrm{d}L}{\mathrm{d}\dot{\mathbf{q}}}-\frac{\delta L}{\delta \mathbf{q}}=\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\delta K(\mathbf{q},\dot{\mathbf{q}})}{\delta\dot{\mathbf{q}}}\right)-\frac{\delta K(\mathbf{q},\dot{\mathbf{q}})}{\delta\mathbf{q}}+\frac{\delta P(\mathbf{q})}{\delta\mathbf{q}}=\boldsymbol{\tau}; \tag{12}$$

čia: $\boldsymbol{\tau}$ – apibendrintose koordinatėse veikiančių jėgų matrica, $K(\boldsymbol{q},\boldsymbol{q}\text{`})$ – kinetinė energija, $P(\boldsymbol{q})$ – potencinė energija.

Pakeitus apibendrintąsias koordinates absoliučiosiomis Dekarto koordinatėmis, gaunama tokia Lagranžo formuluotės išraiška:

$$\mathbf{J}^T(\mathbf{X})F=\mathbf{M}(\mathbf{X})\ddot{\mathbf{X}}+\mathbf{C}(\mathbf{X},\dot{\mathbf{X}})\dot{\mathbf{X}}+\mathbf{G}(\mathbf{X}); \tag{13}$$

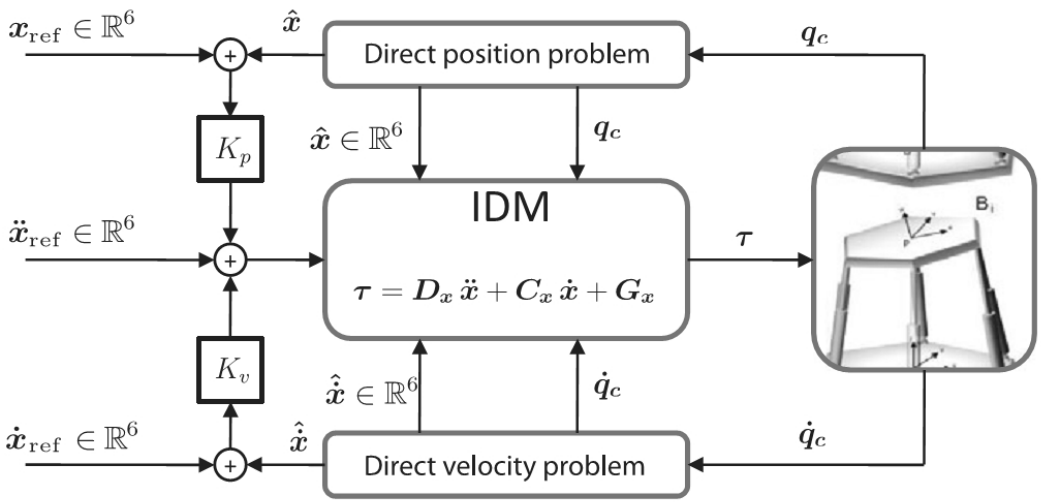
čia: $\boldsymbol{J}$ – Jakobiano matrica, $F$ – vykdiklio generuojama jėga veikianti i-tojoje manipuliatoriaus kojoje, kojos sąnaros ašies kryptimi nukreipta į judančią platformą, $\boldsymbol{M}$ – platformos masių matrica, $\boldsymbol{C}$ – Koriolio ir išcentrinių jėgų matrica, $\boldsymbol{G}$ – gravitacinių jėgų matrica.

Niutono-Oilerio metodu išvesta manipuliatoriaus dinamikos lygtis [9]:

$$\boldsymbol{\tau}=\mathbf{M}(\mathbf{q})\cdot\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})+\mathbf{G}(\mathbf{q}); \tag{14}$$

čia: $\boldsymbol{\tau}$ – apibendrintose koordinatėse veikiančių jėgų vektorius, $\boldsymbol{M(q)}$ – platformos masių matrica, $\boldsymbol{C}(\boldsymbol{q},\boldsymbol{q}`)$- Koriolio ir išcentrinių jėgų matrica, $\boldsymbol{G}(\boldsymbol{q})$ – gravitacinių jėgų matrica.

Šias manipuliatoriaus dinamikos lygtis galima naudoti matematiniam platformos modeliui sudaryti, tam geriausiai tinka inžinerinių skaičiavimų programinis paketas MATLAB, bei grafine vartotojo sąsaja paremtas modeliavimų paketas Simulink, kurio aplinkoje modelį galima sudaryti, komponuojant funkcinius blokus ir iš jų sudarant diferencialinėmis lygtimis aprašytą matematinį sistemos modelį arba naudojantis Simulink programinio paketo biblioteka, skirta mechaninėms sistemos modeliuoti – SimMechanics [6]. Savo darbe A. Zubizarreta (2012) [7] naudoja 6-6 konfigūracijos Stiuarto platformos dinamikos lygtis sudaryti manipuliatoriaus modeliu paremtą valdymo sistemą su išplėstomis sukimo momento valdymo galimybėmis – ECTC (*angl*. *Extended Computed Torque Control*). Šios valdymo sistemos modelio blokinė schema pateikta (5 pav.).




5 pav. ECTC valdymo sistemos modelio blokinė schema (A. Zubizarreta 2012)

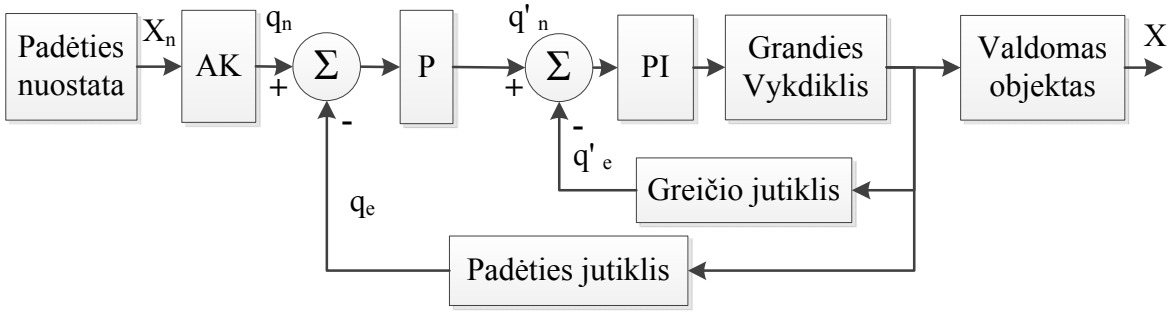
## 1.5 Modelių ir valdymo sistemų apžvalga

### 1.5.1 Valdymo sistemos

Valdymo sistemos tikslas (užduotis) yra užtikrinti manipuliatoriaus užduoties vykdymą su didžiausiu įmanomu tikslumu, atsižvelgiant į statines ir dinamines nuokrypas, trikdžių kompensavimą [11]. Stiuarto platforma, kaip ir kiekvienas manipuliatorius ar mechanizmas, yra valdoma savo grandžių apibendrintųjų koordinačių sistemoje arba absoliučioje koordinačių sistemoje, šios dvi koordinačių sistemos surištos kinematinėmis lygtimis. Todėl, norint modeliuoti ir valdyti Stiuarto platformą, reikia valdymo sistemoje spręsti kinematinius uždavinius minėtosioms koordinačių sistemų reikšmėms apskaičiuoti. Paprasčiausias būdas

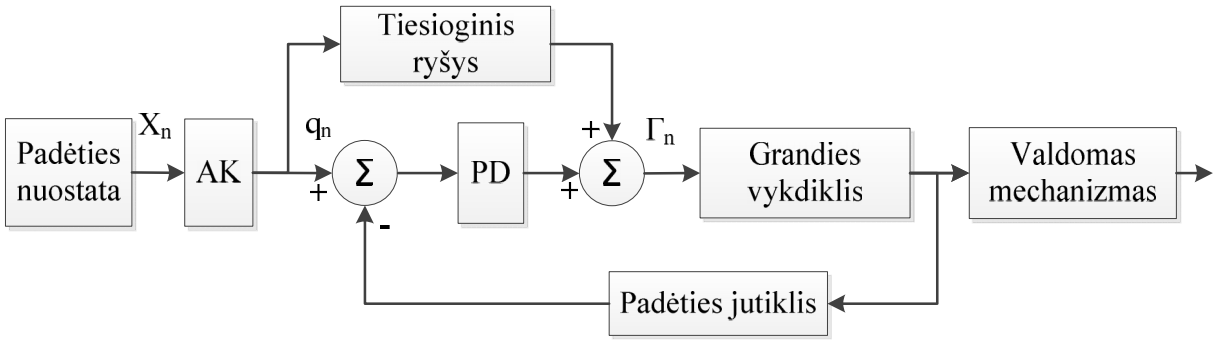
valdyti Stiuarto platformą Dekarto koordinačių sistemoje yra atvirkštinio kinematikos uždavinio sprendimas. Šios nesudėtingos judesio valdymo sistemos dažniausia yra pagrįstos tiesiniu pavienių grandžių valdymu (6 pav.), naudojant proporcinius – P, integralinius – I ir diferencialinius - D reguliatorius, bei jų kompleksines atmainas. Tokias valdymo sistemas galima išskirti į dvi klasikines grupes: greičio (padėties) ir jėgos valdymo.

Įprastai greičio valdymui naudojamas PI reguliatorius su padėties grįžtamuoju ryšiu. Taigi greičio reguliavimas leidžia kontroliuoti apkrovos švytavimus ir padėties grįžtamasis ryšys kontroliuoja dinaminį atsaką (nusistovėjimo laiką). Tačiau toks valdymo dėsnis nėra visiškai tinkamas, kai reikalaujama dinaminio tikslumo, ypač kai dinaminis trajektorijos sekimas turi būti tikslus. Grandies judesio greičio reguliatorius optimaliai išnaudojamas pereinamojo proceso metu, kai grįžtamojo ryšio atsakas yra lėtas.




6 pav. Grandies greičio valdymo schema

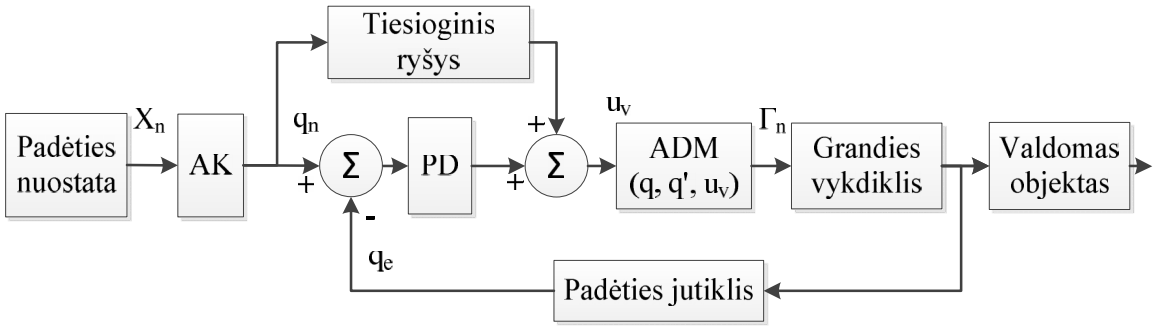
Tiesinis proporcinis-diferencialinis PD jėgos valdymas, kartais dar vadinamas apskaičiuotųjų momentų valdymu (7 pav.), pasižymi pagerintomis dinaminio tikslumo savybėmis pereinamųjų procesų metu, lyginant su greičio valdymo sistema. Statinis tikslumas apibendrintųjų grandžių valdymo sistemoje užtikrinamas jos neigiamojo padėties grįžtamojo ryšio, o absoliučioje Dekarto koordinačių sistemoje – mechanizmo specifinės geometrinės struktūros. Dinaminis valdymo sistemos tikslumas užtikrinamas tiksliais grandies parametrų koeficientais, bei reguliatoriaus derinimu. Tokios valdymo sistemos dažnai naudojamos dėl savo paprasto pritaikymo, derinimo ir atitinkamo tikslumo. Tačiau šios valdymo sistemos silpnoji vieta yra įvairūs pašalinių jėgų sukelti trikdžiai: gravitacijos jėga, sausosios trinties jėga, bei neįvertintos apkrovos jėgos, susijusios su manipuliatoriaus darbo eiga ( pvz.: frezavimu, gręžimu ir pan.).




7 pav. Linijinio grandžių momento valdymo schema

Netiesinės valdymo sistemos yra taikomos, kai tiesinių valdymo sistemų nepakanka užtikrinti dinaminiam manipuliatoriaus tikslumui. Jos naudoja išvestus dinaminius mechanizmo matematinius modelius valdymo grandyje (8 pav.). Vykdiklio valdiklis gali būti paremtas tiesiniu PID reguliatoriumi komponuojamu su prediktyviu, adaptyviu ar kitu sudėtingesniu reguliatoriumi.

Viena klasikinių valdymo sistemos topologijų yra apskaičiuotųjų momentų valdymo sistema (angl. *CTC – Computed Torque Control*). Ši valdymo sistemos topologija naudoja roboto atvirkštinės dinamikos lygčių matematinį modelį, apskaičiuoti reikalingoms jėgoms roboto vykdymo grandyse. Manipuliatoriaus dinaminio modelio panaudojimas taip pat leidžia įvertinti ir kompensuoti anksčiau jau minėtas, linijinėse valdymo sistemose neįvertintas jėgas.




8 pav. Netiesinės valdymo sistemos pavyzdys

### 1.6 Pritaikymo sričių analizė

Lygiagrečiosios kinematinės grandinės, priešingai nei nuosekliosios kinematinės grandinės robotams, apkrova darbiniame taške pasiskirsto daugmaž tolygiai, taip pat lygiagrečios konstrukcijos standumas yra žymiai didesnis nei nuosekliosios. Šios ir kitos teigiamos lygiagrečiųjų robotų savybės leidžia santykinai nedideliems lygiagretiesiems robotams manipuliuoti didelėmis apkrovomis. Originalūs pirminiai lygiagrečiųjų robotų pritaikymai buvo:

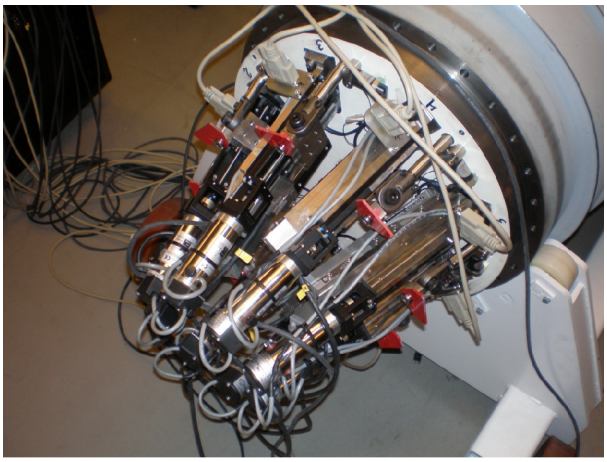
- Stiuarto platforma – skrydžio stimuliatorius;

- Gofo manipuliatoriaus – pozicionavimo sistema padangų nusidėvėjimui tirti.

Praėjus keliems dešimtmečiams ir lygiagrečiųjų robotų mokslui žengiant į priekį, atsirado ir daug naujų pritaikymo sričių, skirtų šiandieniniams technologiniams poreikiams. Viena plačiausių pritaikymo sričių yra pozicionavimo sistemos, kurios gali būtų įgyvendintos labai plačiame skalių diapazone – nuo nanometrų iki, nūdienai įprastų, milimetrinių skalių. Pozicionavimą nanometrų skalėje leidžia atlikti tokios naujos technologijos kaip pjezoelektriniai vykdikliai, didelio tikslumo elektromechaniniai vykdikliai ar pneumatiniai-hidrauliniai vykdikliai su didelės skiriamosios gebos grįžtamojo ryšio jutikliais.

### 1.6.1 Tikslus pozicionavimas

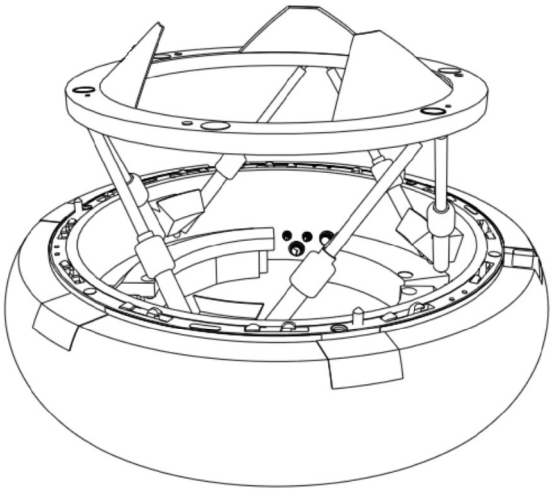
Rui Chen savo darbe [12] pritaiko Stiuarto platformą medicininiam pozicionavimo įrenginiui (9 pav.), galinčiam pozicionuoti įrankį 10 μm tikslumu 10 mm spindulio darbinėje erdvėje, įrankio apkrova 10N. Naudojami elektromechaniniai servo vykdikliai, manipuliatorius valdomas apibendrintųjų grandžių srityje naudojant atvirkštinį padėties uždavinį, bei įvertinant vykdiklių apkrovas.



9 pav. Pozicionavimo įrenginys

### 1.6.2 NASA LIDS (*low impact docking system*)

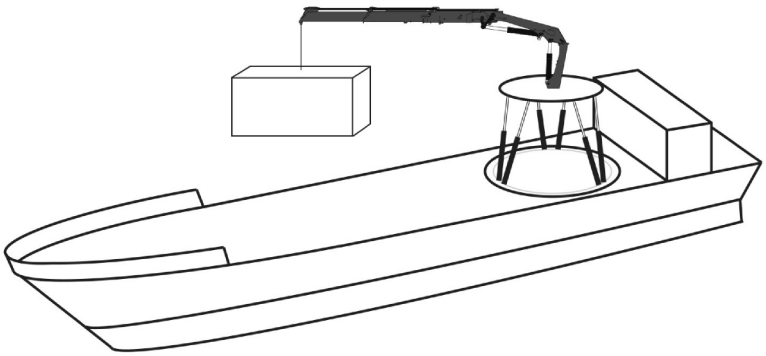
Tai Amerikos Nacionalinės aeronautikos ir kosmoso administracijos sukurta kosminių sistemų švartavimo sistema, paremta šešių laisvės laipsnių lygiagretaus manipuliatoriaus veikimo principu (10 pav.). Manipuliatoriaus geometrija artima 3-3 konfigūracijos Stiuarto platformai. Manipuliatorius gali būti valdomas autonominiu arba rankiniu būdu. Naudojami elektromechaniniai vykdikliai, bei apkrovos jutikliai [13].



10 pav.LIDS švartavimosi sistema

### 1.6.3 Bangų stabilizavimo sistema laivo kranui

Sistema išnaudoja vieną iš praktiškiausių lygiagrečių manipuliatorių pritaikymo sričių – judesio stabilizavimą, didelis išvystomos jėgos – manipuliatoriaus masės santykis, tikslumas ir greitis pritaikomi sklandžiam krovinio pakrovimui ar iškrovimui esant net ir dideliam bangavimui (bangos aukštis nuo 2,5 iki 4 m.). Atsakingam manipuliatoriaus valdymui pritaikyta ir optimizuota valdymo sistema su manipuliatoriaus dinaminiu modeliu [14]. Sistemos principinė iliustracija pateikta (11 pav.).



11 pav.Sistemos principo iliustracija

### 1.6.4 Lygiagrečios kinematinės struktūros kranai (*RoboCrane*)

Apverstos Stiuarto platformos geometrijos manipuliatoriai (12 pav.), kurių vykdikliai yra gervės su servo varikliais. Turi šešis laisvės laipsnius, leistina apkrova iki 855 kg, judėjimo greitis – 3 cm/s. Pritaikyta nesudėtinga grandžių valdymo sistema su PID reguliatoriais, visos sistemos greitaveika, nuo komandos suformulavimo iki judesio įvykdymo, yra apie 20 Hz [15].


12 pav.*RoboCrane* sistema, pastatyta Nacionaliniame standartų ir technologijų institute

## 1.7 Skyriaus išvados

1. Stiuarto platforma – lygiagrečios kinematinės grandinės mechanizmas, kurio geometrija, padėtis erdvėje gali būti aprašytos naudojant tiesioginės ir atvirkštinės kinematikos lygtis (uždavinius). Tiesioginės kinematikos uždaviniai lygiagretiesiems mechanizmams yra sudėtingos netiesinių lygčių sistemos, kurios tik kai kuriais atvejais gali būti išsprendžiamos, gaunant nuo 12 iki 40 realiųjų sprendinių.
2. Dėl pirmoje išvadoje aprašytos priežasties, Stiuarto platformų valdymui daug dažniau naudojamos atvirkštinės kinematikos lygtis, leidžiančios valdyti mechanizmą jo apibendrintųjų grandžių (vykdiklių) koordinačių sistemose.
3. Stiuarto platformos mechanizmo dinamiką apibūdinančios lygtys gali būti išvedamos keliais būdais: Lagranžo, Niutono-Oilerio ir Keino metodais. Tačiau, dinaminis mechanizmo modelis neapsieina be dalinio tiesioginio greičių uždavinio sprendimo, naudojant vykdiklių ir judančios platformos judėjimo greičių išvestinių matricą – vadinamąjį Jakobianą, kurio skaičiavimas yra pakankamai sudėtingas, todėl gali apriboti tokio modelio pritaikymą realaus laiko valdymo sistemose.
4. Mechanizmo valdymo sistemoms, kai nebūtina įvertinti mechanizmo dinaminių apkrovų įtakos, dažniausiai naudojamos nesudėtingos manipuliatoriaus grandžių apibendrintųjų koordinačių apskaičiavimo lygtys, sprendžiant atvirkštinį padėčių ir greičių uždavinį.

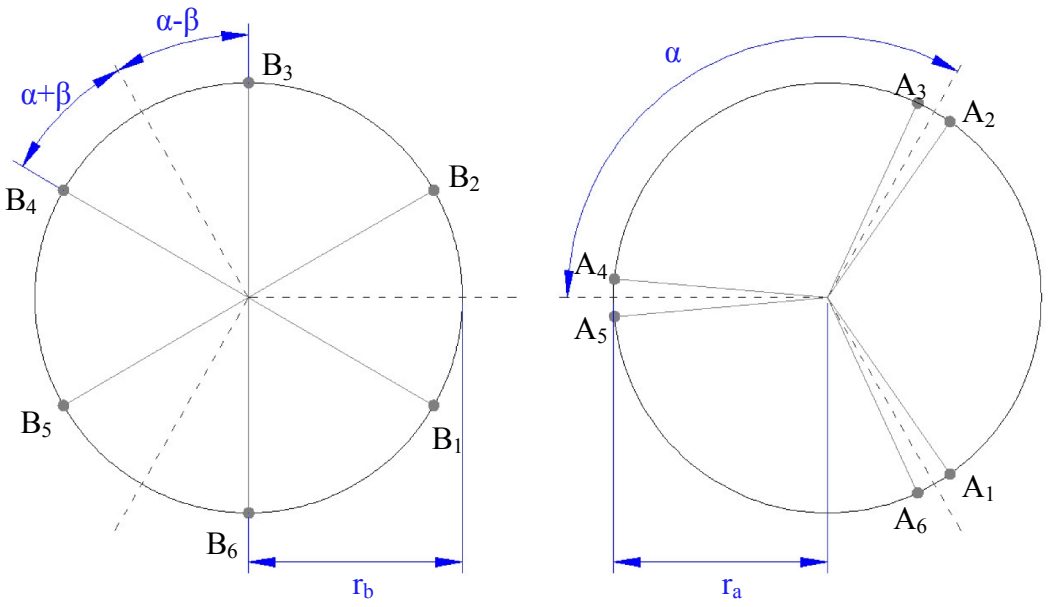
# 2. METODOLOGINĖ DALIS

## 2.1 Manipuliatoriaus parametrų nustatymas

Dalinis Stiuarto platformos kinematinis modelis sudaromas, naudojant nesudėtingas atvirkštinės kinematikos lygtis tam, kad būtų išvengta netiesinių tiesioginio kinematikos uždavinio lygčių sprendimo. Šis modelis leidžia apskaičiuoti manipuliatoriaus vykdiklių (kojų) ilgius, kai žinoma manipuliatoriaus judančios platformos padėtis absoliučioje koordinačių sistemoje. Atvirkštinės kinematikos lygčių modelis dažniausia naudojamas manipuliatoriaus trajektorijos generavimo stadijoje, iš norimos erdvinės padėties apskaičiuojant užduotis vykdikliams. Atvirkštinės kinematikos lygčių modeliui užduotis gali būti formuojama kaip platformos padėties vektorius **X,** susidedantis iš trijų linijinių poslinkių ir trijų ašinių posūkių erdvėje (15).

$$\mathbf{X} = [x, y, z, \psi, \theta, \varphi]. \tag{15}$$

Pagrindiniai (pradiniai) modelio parametrai yra pagrindo, bei judančios platformų apskritimų spinduliai - $r_b$, $r_a$; judančios platformos aukštis – $h$, atraminių taškų išdėstymo kampas $\alpha$ = 120°, atitraukimo kampai (*angl. Offset*) atraminiams taškams $\beta$ = [0° ; 30°], bei judančios platformos atraminių taškų pasukimo kampas $\gamma$, dažniausia 60° (13 pav.).




13 pav.Platformų taškų išdėstymo pavyzdys, $\alpha$ = 120°, kairėje – $\beta$=30°, dešinėje - $\beta$=5°

Priklausomai nuo pasirinkto atitraukimo kampo $\beta$ pagrindo ir judančios platformos atraminiams taškams išdėstyti, gaunama viena iš trijų geometrinių konfigūracijų: 3-3, 3-6 ar 6-6. Judančios platformos atraminių taškų padėtis koordinačių pradžios atžvilgiu apskaičiuojama pagal:

$$\mathbf{a}_i = \mathbf{p}_i \cdot \mathbf{R} + \mathbf{T}; \tag{16}$$

čia: $\boldsymbol{a_i}$ –judančios platformos *i*-tasis atramos taškas, platformos koordinačių sistemoje, $\boldsymbol{p}_i$ – *i*-tojo atraminio platformos taško vektorius absoliučioje koordinačių sistemoje, $\boldsymbol{R}$ – Oilerio kampų posūkių matrica, $\boldsymbol{T}$ - perkėlimo vektorius.

Platformos erdvinei orientacijai nusakyti naudojama 6x6 dydžio posūkio matrica $\boldsymbol{R}$, sudaryta iš Oilerio kampų (posūkių) aplink $O_Z$, $O_Y$ ir $O_X$ ašis atitinkamai (17), nurodanti judančios platformos koordinačių sistemos posūkį, lyginant su absoliučios koordinačių sistemos pradžios tašku.

$$\mathbf{R} = \begin{bmatrix} cos\psi\, cos\theta & sin\psi\, sin\theta\, sin\varphi - sin\psi\, sin\varphi & cos\psi\, sin\theta\, cos\varphi + sin\psi\, sin\varphi \\ sin\psi\, cos\theta & sin\psi\, sin\theta\, sin\varphi + cos\psi\, cos\varphi & sin\psi\, sin\theta\, cos\varphi - cos\psi\, sin\varphi \\ -sin\theta & cos\theta\, sin\varphi & cos\theta\, cos\varphi \end{bmatrix}; \tag{17}$$

čia: $\psi$,$\theta$ ir $\varphi$ – elementarieji posūkiai apie $O_Z$, $O_Y$ ir $O_X$ ašis atitinkamai.

Judančios platformos koordinačių sistemos O', esančios jos gravitacijos centre, padėtis erdvėje nusakoma 3x1 dydžio perkėlimo vektoriumi $\boldsymbol{T}$, jį sudaro elementarieji postūmiai x, y ir z ašimis (18). Perkėlimo vektorius nurodo judančios platformos koordinačių sistemos pradžios taško padėtį absoliučioje koordinačių sistemoje (3 pav.).

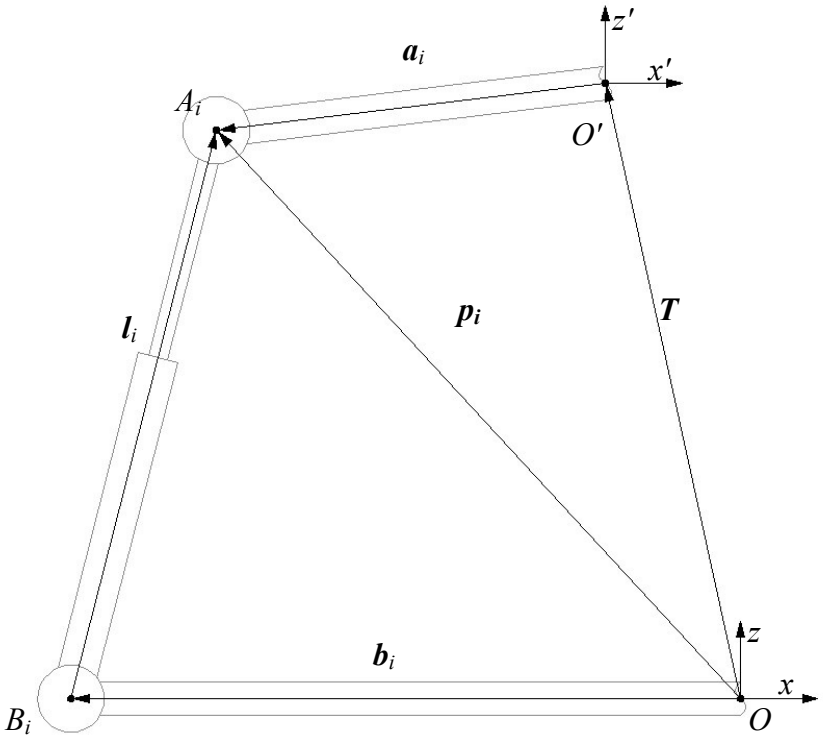
$$\mathbf{T} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}; \tag{18}$$

čia: $T_x$, $T_y$ ir $T_z$ – elementarieji postūmiai *x*, *y* ir *z* ašimis atitinkamai.

Žinant visus anksčiau paminėtus dydžius *i*-tosios kojos vektorius apskaičiuojamas pagal formulę:

$$\mathbf{l}_i = \mathbf{T}_i + \mathbf{R} \cdot \mathbf{a}_i - \mathbf{b}_i; i = 1,..,6.; \tag{19}$$

čia: $\boldsymbol{l}_i$ – *i*-tosios kojos vektorius, $\boldsymbol{b}_i$ – *i*-tojo platformos taško vektorius.

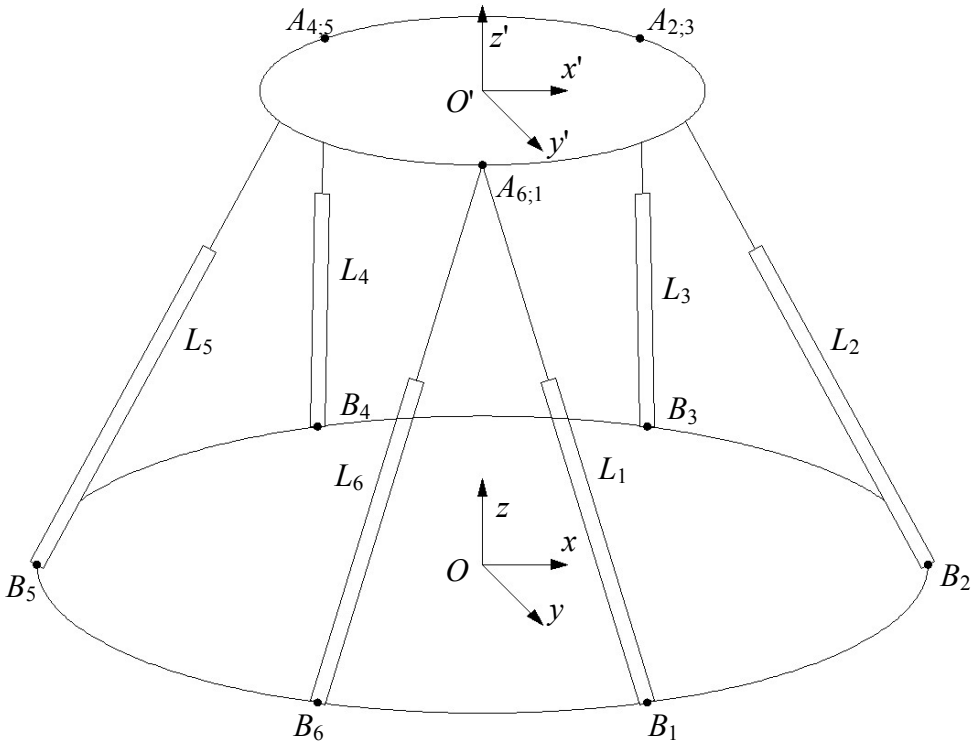



14 pav.Stiuarto platformos geometriją apibrėžiantys vektoriai

Naudojantis aprašytomis lygtimis, MATLAB programavimo aplinkoje sukuriama programa, apskaičiuojanti judesio modeliavimui reikalingus pradinius geometrinius parametrus pagal pageidaujamo vektoriaus $\boldsymbol{X}$ duomenis (trajektoriją). Funkcijos kodas pateiktas 2 priede.

Pradiniame modelio sudarymo ir tyrimo etape nebus žinomas platformos aukštis, kadangi jį reikalinga surasti pagal reikalingų vykdiklių parametrus, todėl MATLAB aplinkoje parašyta programa, kuri naudojama kaip funkcija pagrindinėje programoje, ši funkcija randa platformos aukštį pagal duotą reikalingą vykdiklio ilgį, funkcijos kodas pateiktas 3 priede.

Funkcija parašyta įvertinant, kad pradinėje padėtyje manipuliatoriaus judančios platformos centras yra toje pačioje tiesėje, kaip ir pagrindo platformos bei abiejų platformų $z$ ašys (15 pav.). Taip pat platforma yra nepasukta erdvėje pagrindo atžvilgiu, todėl posūkio matrica $\boldsymbol{R}$ yra lygi vienetinei 3x3 dydžio matricai. Šios prielaidos leidžia apskaičiuoti aukštį, naudojant Pitagoro teoremą, šiuo atveju įžambinė – vykdiklio ilgis, o statinis – vektorių $\boldsymbol{a}_i$ ir $\boldsymbol{b}_i$ skirtumo $\boldsymbol{q}_0$ ilgis.



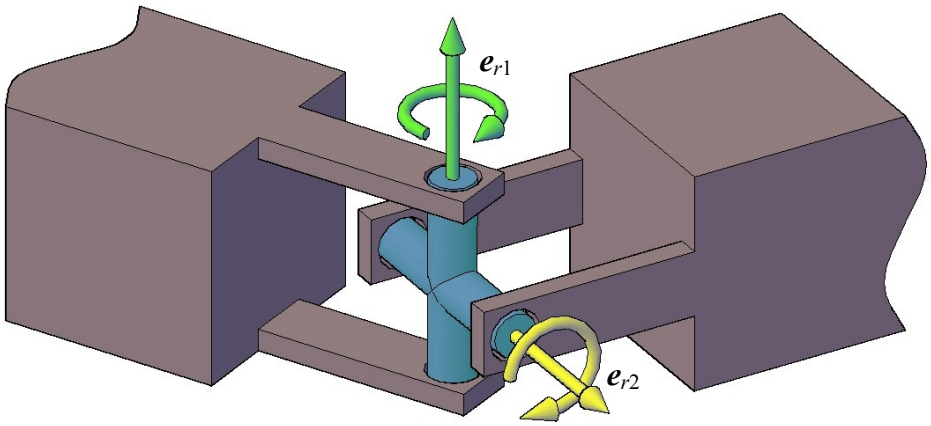

15 pav.3-6 tipo Stiuarto platformos geometrija .

Manipuliatoriaus dinamika bus modeliuojama naudojantis MATLAB Simulink SimMechanics funkcinių blokų biblioteka. Norint panaudoti minėtosios bibliotekos funkcinius blokus, reikia rasti pradinius modelio dinamikai apskaičiuoti reikalingus dydžius:

- pagrindo ir platformos universaliųjų sąnarų posūkių ašis;
- manipuliatoriaus kojų vektorius, jų ilgius ir ortus;
- hidraulinio cilindro ir koto:
    - ilgių vektorius;
    - gravitacijos centų vektorius;
    - krypties (posūkio) vektorius;
- judančių dalių (cilindro korpusų, kotų ir judančios platformos) mases ir inercijos momentus.

Manipuliatoriaus dinaminis modelis buvo sudarytas naudojantis SimMechanics funkcinių blokų biblioteka, remiantis [6] straipsnyje pateikta informacija. Parametrų apskaičiavimo lygtys taip pat sudarytos remiantis minėtuoju straipsniu. Universaliųjų sąnarų ašių ortai apskaičiuojami dviejuose taškuose kiekvienai *i*- tajai kojai:

1. $B_i$ – pagrindo atraminiame taške;
2. $A_i$ – platformos atraminiame taške.

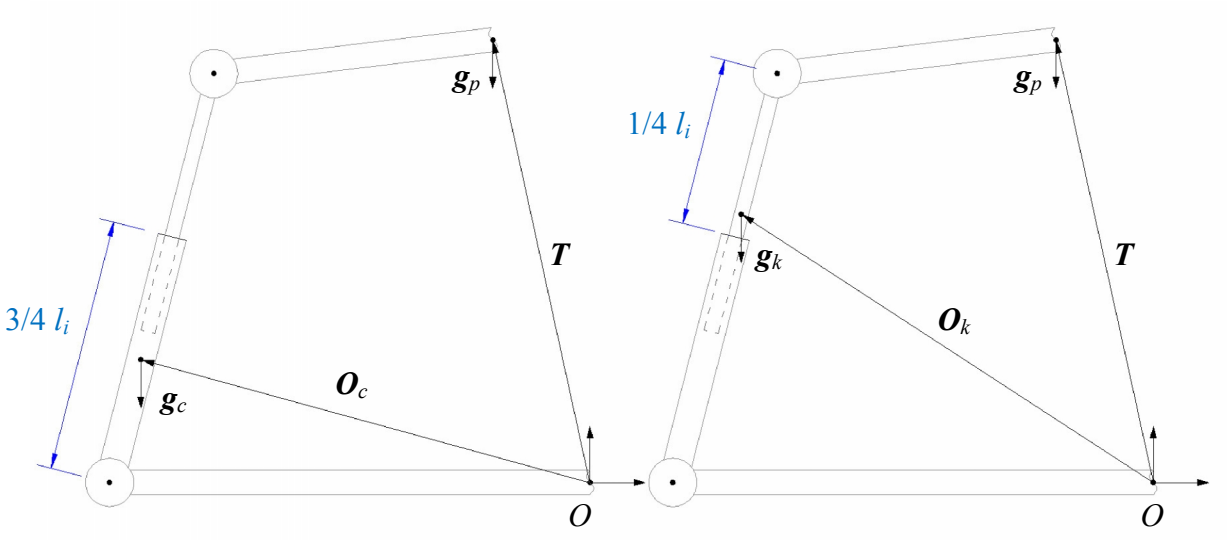



16 pav.Universaliosios sąnaros principinė schema

Universalioji (dar vadinama Huko arba kardaninė sąnara) turi du rotacinius laisvės laipsnius aplink dvi, statmenas viena kitai, ašis (16 pav.). Sąnaros pirmosios ašies ortas apskaičiuojamas iš su sąnara sujungtos manipuliatoriaus kojos ir Dekarto koordinačių sistemos z ašies vektorių vektorinės sandaugos, todėl gautasis ortas yra statmenas minėtųjų sandaugos ašių sudaromai plokštumai. Antrosios ašies ortas gaunamas iš pirmosios ašies orto ir su sąnara sujungtos kojos vektorių vektorinės sandaugos, todėl gautasis antrosios ašies ortas yra statmenas pirmosios ašies ortui ir su sąnara sujungtos kojos vektoriui. Universaliųjų sąnarų taškuose $B_i$ ir $A_i$ ortai identiški. Aprašytus veiksmus atliekantis MATLAB funkcijos kodas pateiktas 4 priede.

Manipuliatoriaus kojų ilgių vektoriai apskaičiuojami pagal jau minėtąjį atvirkštinį padėčių uždavinį: žinant judančios platformos atraminių taškų vektorius $a$ ir pagrindo atraminių taškų vektorius $b$, kojų ilgio vektorius $l$ bus atitinkamai pirmojo ir antrojo vektorių skirtumas. Kojos vektoriaus ortas $e_l$ gaunamas padalinus kojos ilgio vektorių iš jo ilgio skaliarinės vertės. MATLAB funkcijos kodas skirtas kojų parametrams rasti pateiktas 5 priede.

Norint apskaičiuoti cilindro korpuso ir koto parametrus (gravitacijos centrus, mases ir kt.), nominalus kojos vektoriaus ilgis dalinamas į santykinius ilgius (17 pav.), atitinkančius cilindro ir koto ilgių vektorius pradinėje manipuliatoriaus pozicijoje, kai $\boldsymbol{T} = [0\ 0\ h]$ ir $\boldsymbol{R} = \boldsymbol{I}_{3x3}$. Numatytieji kojos ilgio vektoriaus santykiai: cilindro ilgio vektorius - 3/4 kojos ilgio vektoriaus ir koto ilgio vektorius – 1/4 kojos ilgio vektoriaus.




17 pav.Manipuliatoriaus dinaminių parametrų vektoriai

Gravitacijos centrų vektoriai apskaičiuojami panašiai kaip ir cilindro korpuso, bei koto ilgių vektoriai – naudojant visos kojos santykinius ilgius. Cilindro korpuso gravitacijos centro vektorius atitinka 3/8 manipuliatoriaus kojos ilgio vektoriaus dalį, o koto gravitacijos centras – 5/8 kojos ilgio vektoriaus. Parašytos MATLAB funkcijos kodas pateiktas 6 priede.

Dinaminio manipuliatoriaus modelio judančių dalių (cilindro korpuso, koto ir platformos) masės ir inercijos apskaičiuojamos naudojantis MATLAB Stiuarto platformos simuliacijos paketo funkcija *inertiaCylinder*, kuri priima argumentus: tankį, aukštį (ilgį) ir išorinį, bei vidinį (jei reikalingas tuščiaviduris cilindras, antraip lygus 0) diametrus ir apskaičiuoja duotųjų parametrų cilindro masę ir inercijos momentą.

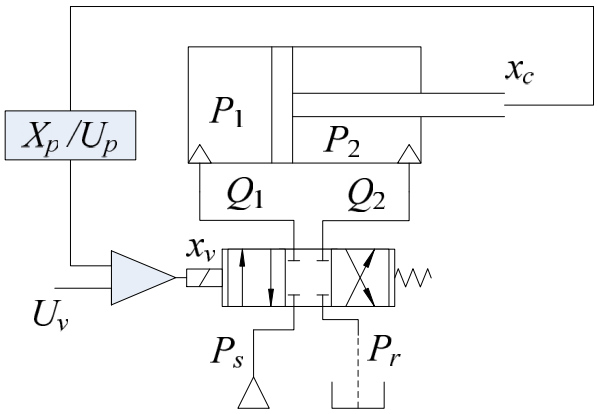
## 2.2 Manipuliatoriaus vykdiklių modeliavimas

Hidraulinės servo sistemos pasižymi geromis išvystomos jėgos ir sistemos masės santykio, bei tikslumo charakteristikomis, tačiau šių sistemų dažninės charakteristikos yra prastesnės nei mechaninių ar pneumatinių sistemų. Palyginus hidraulines servo pavaras su panašiomis elektromechaninėmis ar pneumatinėmis paaiškėjo, kad pastarosios pranašesnės nedideliu išvystomos galios ir kainos santykiu, kai reikalinga galia mažesnė nei kelios dešimtys kilovatų. Tačiau didesnės galios mechanizmų nišoje hidraulinės servo sistemos pralenkia pastarąsias [12-14]. Matematinis hidraulinių sistemų modelis sudaromas iš hidraulinio cilindro ir servo vožtuvo lygčių sistemų, susietų hidraulinio skysčio debitais $Q_1$ ir $Q_2$ (kur $Q_1$ – debitas į cilindro kamerą be koto; $Q_2$ – debitas į cilindro kamerą su kotu) [17]. Cilindrui su vienu kotu (18 pav.) debitai bei slėgiai skirtingose kameros pusėse skiriasi nuo to, į kurią pusę juda cilindro kotas. Debitai išreiškiami lygčių sistemomis [18]:

$$Q_1 = \begin{cases} C_q\, w\, x_v \sqrt{\dfrac{2}{\rho}(p_s - p_1)}, & x_v \geq 0; \\ C_q\, w\, x_v \sqrt{\dfrac{2}{\rho}(p_1 - p_r)}, & x_v < 0; \end{cases} \tag{20}$$

$$Q_2 = \begin{cases} C_q\, w\, x_v \sqrt{\dfrac{2}{\rho}(p_2 - p_r)}, & x_v \geq 0; \\ C_q\, w\, x_v \sqrt{\dfrac{2}{\rho}(p_s - p_2)}, & x_v < 0; \end{cases} \tag{21}$$

čia: $C_q$ – debito koeficientas įvertinantis slėgio kritimą proporciniame vožtuve; $w$ – vožtuvo ploto gradientas, m; $x_v$ – vožtuvo sklendės padėtis, m; $\rho$ – hidraulinio skysčio tankis, $m^3/kg$; $p_1$ ir $p_2$ – atitinkamai slėgiai cilindro kameroje be koto ir su kotu, Pa; $p_s$ – darbinis slėgis, Pa; $p_r$ – skysčio rezervuaro slėgis, Pa.




18 pav.Hidraulinės servo sistemos principinė schema

Hidraulinį cilindrą veikiančios jėgos, sukeltos skysčio slėgio, aprašomos hidraulinio skysčio spūdumą įvertinančiomis diferencialinėmis slėgio lygtimis [16]:

$$\frac{\mathrm{d}p_1}{\mathrm{d}t} = \frac{\beta_S}{V_1}(Q_1 - A_1 v); \tag{22}$$

$$\frac{\mathrm{d}p_2}{\mathrm{d}t} = \frac{\beta_S}{V_2}(A_2 v - Q_2); \tag{23}$$

čia: $\beta_S$ – skysčio spūdumo modulis, Pa; $V_1$ ir $V_2$ – cilindro kamerų dalių tūriai, $m^3$; $A_1$ ir $A_2$ – stūmoklio darbiniai plotai atitinkamai cilindro kameroje be koto ir su kotu, $m^2$; $v$ – koto greitis, m/s.

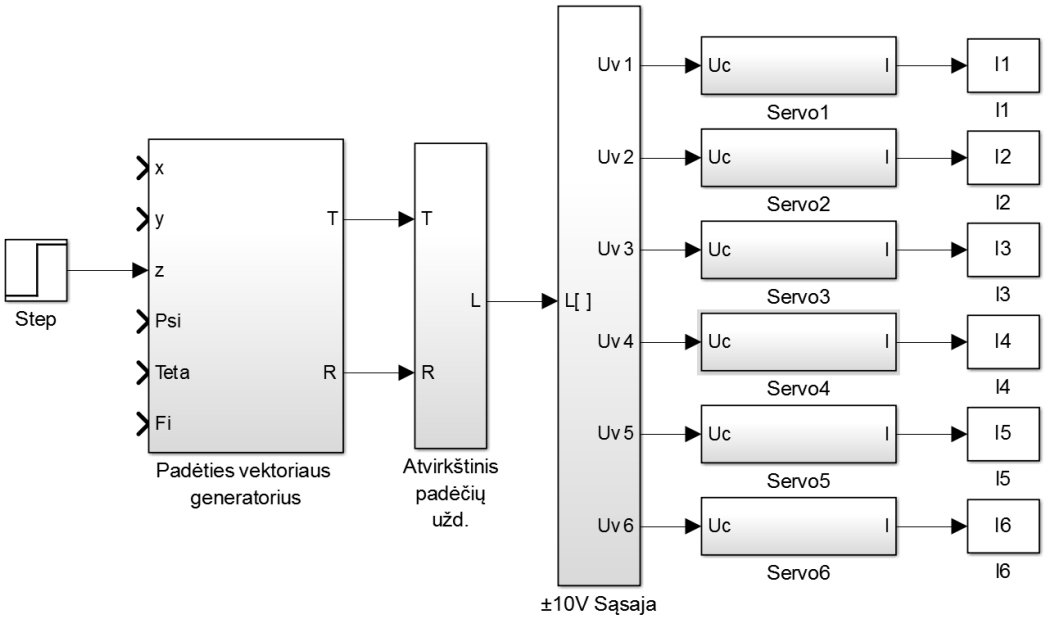
Hidraulinio cilindro koto judesio lygtis (24) randama pasinaudojus Niutono 2-ojo dėsnio lygtimi įvertinus cilindrą veikiančias hidraulinio slėgio sukurtas jėgas:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = p_1 A_1 - p_2 A_2 - F_a; \tag{24}$$

čia: $F_a$ – cilindro apkrovos jėga, N.

### 2.3 . Sistemos modeliavimas Simulink programoje

Stiuarto platformos atviro kontūro valdymo sistema modeliuojama MATLAB Simulink terpėje. Visos išvestosios hidraulinės sistemos matematinio modelio lygtys Simulink terpėje pervedamos į blokinį modelį (19 pav.). Tarp trajektorijos generavimo bloko ir servo mechanizmo įterptas sąsajos keitimo blokas, kuris generuoja ±10 V analoginio valdymo signalo komandą.



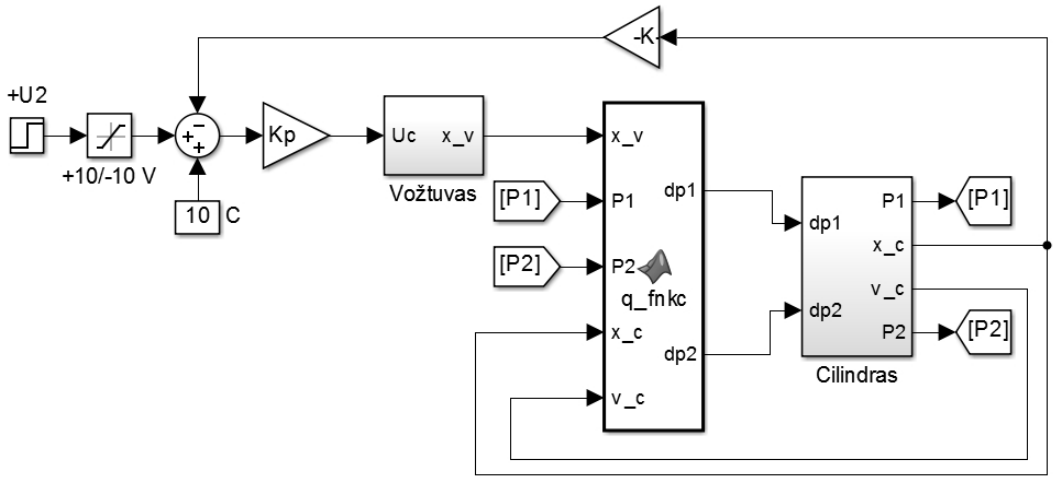

19 pav.Stiuarto platformos apibendrintųjų grandžių valdymo sistema Simulink terpėje

Hidraulinio servo vykdiklio lygtys panaudojamos servo sistemos dinaminiam modeliui Simulink programoje sudaryti (20 pav.). Sudarytasis modelis yra netiesinis, todėl programos simuliacijos parametruose nustatomas pastovaus žingsnio diferencialinių lygčių sprendimo algoritmas, nes kintančio – adaptyvaus žingsnio algoritmas, spręsdamas netiesines diferencialines lygtis, linkęs diverguoti. Modeliuojant buvo naudojami 1 lentelėje pateikti hidraulinės servo sistemos parametrai.

1 lentelė

**Hidraulinės sistemos parametrai**

| Parametras | Vertė | Apibūdinimas |
|---|---|---|
| $A_1$ | 80 mm$^2$ | Pilnas stūmoklio plotas |
| $A_2$ | 35 mm$^2$ | Koto pusės plotas |
| $l$ | 648 mm | Koto ilgis |
| $P_s$ | 160 bar | Sistemos darbinis slėgis |
| $\beta_S$ | 6890 bar | Skysčio spūdumo modulis |
| $w$ | 2,5 mm | Sklendės ploto gradientas |
| $C_q$ | 0,61 | Debito koeficientas |
| $\rho$ | 890 kg/m3 | Hidraulinio skysčio tankis |




20 pav.Hidraulinės servo pavaros matematinio modelio Simulink analogas

## 2.4 Stiuarto platformos Simulink modelis

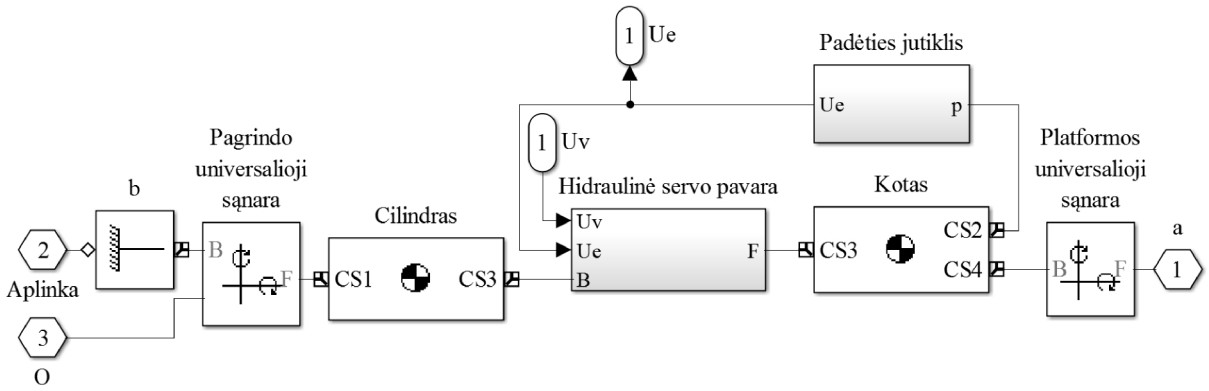
Naudojantis 2.1-2.3 poskyriuose aprašytomis manipuliatorių apibūdinančiomis kinematikos ir dinamikos matematinėmis lygtimis, sukurtas Simulink funkcinių blokų modelis, kuris bus naudojamas manipuliatoriaus judesio modeliavimui.

### 2.4.1 SimMechanics funkcinių blokų modelio sudarymas

SimMechanics funkcinių blokų biblioteka naudojama standžių mechaninių kūnų modeliavimui trimatėje erdvėje. Dvi pagrindinės blokų rūšys SimMechanics bibliotekoje yra mechaniniai kūnai ir sąnaros, būtent iš šių blokų ir sudaromi kompleksiniai mechanizmų modeliai. Kūnų funkciniai blokai naudoja tokius parametrus kūnui apibūdinti:

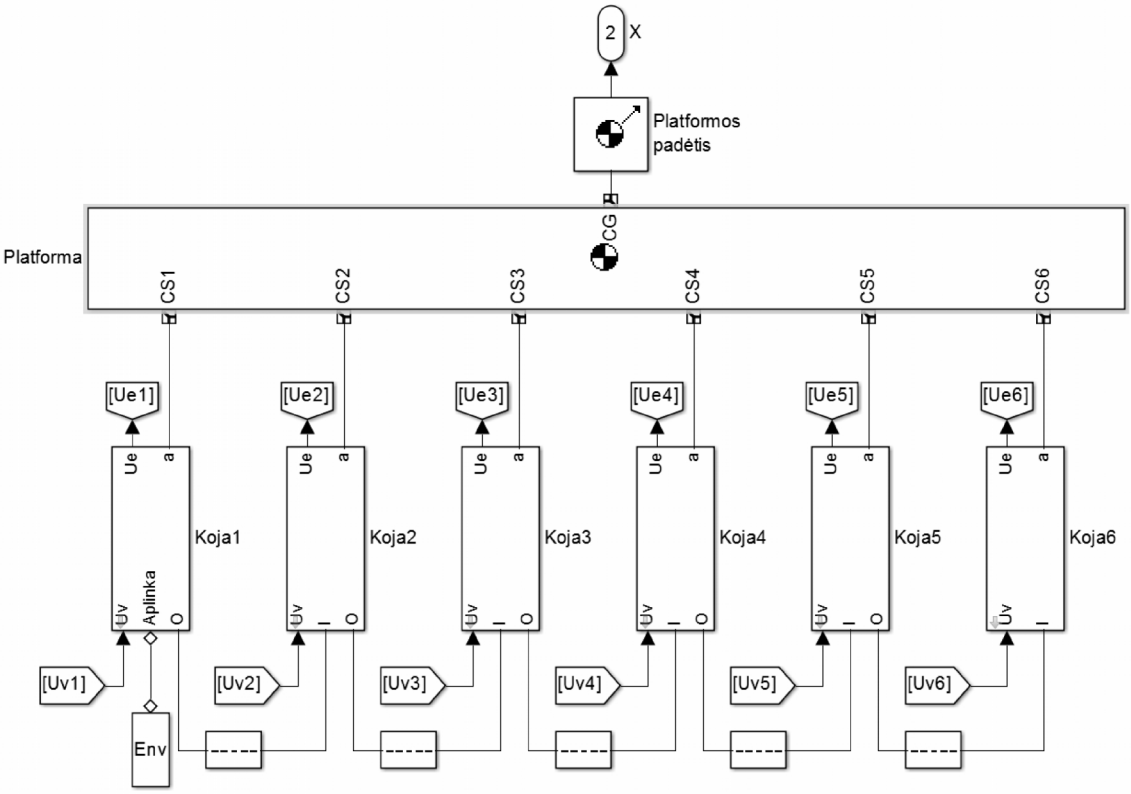
- kūno masė, kg;
- kūno inercijos momentas, kg·m$^2$;
- kūno gravitacijos centras, [x, y, z];
- kūno taškai:
    - taškų koordinačių sistemos CS, [x, y, z];
    - taškų koordinačių sistemų CS, posūkio (orientacijos) matricos $R_{3\times3.}$

Pirmieji trys argumentai yra bendri visam kūnui, o standžiai susietų kūno taškų gali būti daugiau nei du, paprastai jie naudojami surišti kūnus su sąnaromis, ir, jei reikalinga, apibūdinti kūno geometrinę formą. Stiuarto platformos kojos SimMechanics modelis sudaromas iš dviejų standžių kūnų, atstojančių cilindro korpusą ir jo kotą, bei trijų sąnarų: dviejų universaliųjų jungiančių cilindrą su pagrindu, bei su judančia platforma ir vienos cilindrinės sąnaros, tarp cilindro korpuso ir koto. Minėtoji kojos mechaninė struktūra papildoma hidraulinio cilindro matematinio modelio, Simulink blokiniu analogu, sudarytu 2.3 poskyryje (21 pav.). Jis įterptas tarp cilindro ir koto kūnų, bei sutalpintas į posistemę „Hidraulinė servo pavara“.




21 pav.Stiuarto platformos kojos modelis

Sujungus šešis kojų modelius naudojantis standaus kūno bloku, atstojančiu judančiąją platformą, gaunamas pilnas Stiuarto platformos mechanizmo modelis, sudarytas iš SimMechanics funkcinių blokų (22 pav.). Taip pat mechanizmo aplinkai nurodomas laisvojo kritimo pagreičio vektorius – [0, 0, -9,81], kadangi pagal nutylėjimą sistema modeliuoja mechanizmą laisvojo kritimo pagreičiui veikiant neigiamos *y* ašies kryptimi.




22 pav.Stiuarto platformos mechaninės dalies modelis.

## 2.5 Valdomo objekto dinaminių parametrų radimas

Sudarytam Stiuarto platformos mechanizmo modeliui reikia nustatyti jo pereinamojo proceso – vykdiklio eigos pokyčio, parametrus: stiprinimo koeficientą $K_{pr}$, laiko pastoviąją $T_0$ ir vėlavimo trukmę $\tau_0$. Laiko pastoviosios ir vėlinimo trukmės apskaičiavimas bus atliekamas dviem metodais: apskaičiuojant liestinę proceso kreivės perlinkimo taške ir randant tiesę, einančią per du proceso kreivės taškus esančius 28,3% ir 63,2% nusistovėjusios pereinamojo proceso kreivės skaitinės vertės. Stiprinimo koeficientas (25) randamas iš pereinamojo proceso nusistovėjusios vertės ir valdomo objekto įėjimo dydžio pokyčio santykio. Modeliuojamos Stiuarto platformos atveju šie dydžiai yra kojos ilgio pokyčio nusistovėjusi vertė ir servo pavaros valdymo įtampos pokytis:

$$K_{pr} = \frac{l_{nusist.}}{\Delta U_v} \tag{25}$$

Skaičiuojant laiko pastoviąją ir vėlinimo trukmę liestinės, proceso kreivės persilenkimo taške metodu, pirmiausia randama pereinamojo proceso išvestinės, šiuo atveju – vykdiklio greičio kreivės didžiausios vertės pereinamojo proceso metu laikas $t_{v\ max}$, tada pereinamojo proceso kreivės persilenkimo taškas bus šios kreivės vertė tuo pačiu laiko momentu $t_{v\ max}$.

Nubrėžus kreivės liestinę rastame taške, šios liestinės susikirtimo taškai su pradine pereinamojo proceso vertės horizontale ir nusistovėjusios vertės horizontale *x* ašyje, gautieji taškų projekcijos laiko ašyje, bus $\tau_{pr}$ ir $T_{nusist}$ taškai atitinkamai. Tada pereinamojo proceso laiko pastovioji (26) apskaičiuojama atitinkamai:

$$T_{pr} = T_{nusist} - \tau_{pr} \tag{26}$$

Skaičiavimas bus atliekamas MATLAB programoje, šiam tikslui parašyta funkcija apskaičiuojanti reikalingus parametrus iš modeliavimo metu gautų duomenų, funkcijos kodas pateiktas 7 priede.

Skaičiuojant $\tau_{pr}$ ir $T_0$ tiesės, einančios per du taškus esančius pereinamojo proceso kreivės 28,3% ir 63,2% nusistovėjusios vertėse, pirmiausia apskaičiuojamos minėtųjų taškų vertės $y_1$=0,283·$y_{nusist}$ ir $y_2$=0,632·$y_{nusist}$, bei randamos jų projekcijos laiko ašyje: $t_1$ ir $t_2$. Iš šių verčių randama tiesė einanti per apskaičiuotuosius taškus, šios tiesės susikirtimo su pradine pereinamojo proceso vertės horizontale ir nusistovėjusios vertės horizontale *x* ašyje taškų projekcijos laiko ašyje, bus $\tau_{pr}$ ir $T_{nusist}$ taškai atitinkamai. Skaičiavimams naudojama MATLAB funkcijos, naudojančios modeliavimo metu gautus duomenis, funkcijos kodas pateiktas 8 priede.

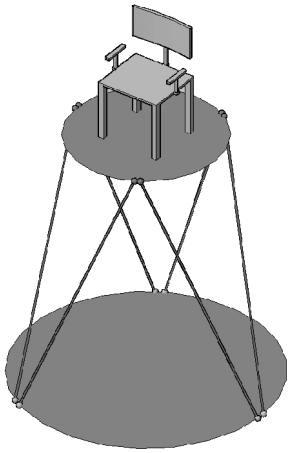
### 2.6 PID reguliatoriaus parametrų nustatymas

Manipuliatoriaus modelio vykdikliai – hidraulinės servo pavaros yra valdomos naudojant PID reguliatorius, todėl jų parametrai bus nustatomi, remiantis literatūroje pateiktais metodais ir reguliatorių derinimo lentelėmis [19]. Naudojami metodai:

- Ziegler ir Nichols šuolinės reakcijos metodas, naudoja du proceso dinamiką apibūdinančius parametrus (stiprinimą *a* ir trukmę $\tau_0$);
- Ziegler ir Nichols šuolinės reakcijos metodo modifikacija pasiūlyta Chien, Hrodes ir Reswick, metodas naudoja tuos pačius dinaminius sistemos parametrus kaip ir originalas;
- Derinimas, naudojant tris dinaminius parametrus – $K_{pr}$, $\tau_{pr}$ ir $T_0$;
- Cohen ir Coon derinimo taisyklės, naudoja tuos pačius dinaminius parametrus, kaip ir derinimo pagal tris parametrus metodas.

# 3. TIRIAMOJI DALIS

Reikalinga parinkti reikiamą geometrinę konstrukciją Stiuarto platformai, kurios šiuo atveju konkreti paskirtis yra ant mobilios platformos esančio žmogaus judesių stabilizavimas arba specifinių judesių simuliavimas. Reikiamos Stiuarto platformos koncepcija pavaizduota 23 pav. Proporcijų palyginimui ant platformos pavaizduota kėdė.



23 pav.Stiuarto platformos koncepcija

Ieškant optimalios geometrinės Stiuarto platformos konfigūracijos, naudojami tokie jos ribiniai darbinės erdvės parametrai:

1. perkėlimo (linijiniai) poslinkiai:
   - vertikalus linijinis poslinkis (*z* ašies kryptimi): ± 200 mm.
   - horizontalūs linijiniai poslinkiai (*x* ir *y* ašių kryptimis): ± 250 mm.
2. ašiniai posūkiai:
   - posūkis apie *z* ašį: ± 45°.
   - posūkis apie *y* ašį: ± 30°.
   - posūkis apie *x* ašį: ± 35°.

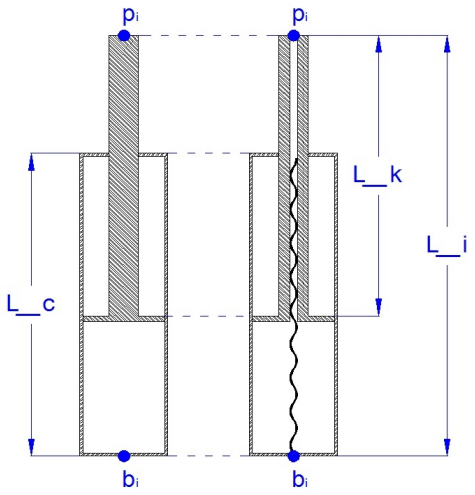
Taip pat pasirenkami pagrindo, bei judančios platformos apskritimų spinduliai, kurie bus naudojami skaičiavimuose:

- pagrindo apskritimo spindulys: 1000 mm.
- judančios platformos apskritimo spindulys: 600 mm.

Pagal duotus parametrus bus ieškoma optimalių kojų ilgių priimant, kad kojos yra elementarūs pneumatinių, hidraulinių ar elektropneumatinių cilindrų modeliai, kuriems galioja tik paprasti jų ilgio apribojimai (24 pav.):

$$\begin{cases} L_{min} = L_C \\ L_0 = L_C + \dfrac{L_K}{2} \\ L_{max} = L_C + L_K \end{cases} \tag{27}$$

čia: $L_{min}$ – mažiausias leidžiamas kojos ilgis, $L_C$ – cilindro korpuso ilgis, $L_0$ – nominalus (vidutinis) kojos ilgis, $L_K$ – koto ilgis, $L_{max}$ – didžiausias leidžiamas kojos ilgis.




24 pav.Elementarūs manipuliatoriaus kojų modeliai

Pagal duotuosius duomenis programa ieškos manipuliatoriaus vykdiklių (kojų) ilgių tarp ribinių verčių, taip patikrinant vykdiklio eigos tinkamumą manipuliatoriaus pritaikymui. Pirmiausia MATLAB programoje suformuojama tikrinamų manipuliatorių eigų seka, parenkant elementus 200 mm žingsniu, o pirmasis elementas parenkamas didesnis nei reikalingas vertikalus mechanizmo poslinkis: „`eiga = [600, 800, 1000, 1200, 1400, 1600, 2000]`“.

Taip pat suformuojama įėjimo kintamųjų vektorių ***X*** seka reikiamoms ribinėms vertėms tikrinti:

```
X = [ -250 ,   0   ,   0   ,   0   ,   0   ,   0   ;
       250 ,   0   ,   0   ,   0   ,   0   ,   0   ;
       0   , -250  ,   0   ,   0   ,   0   ,   0   ;
       0   ,  250  ,   0   ,   0   ,   0   ,   0   ;
       0   ,   0   , -200  ,   0   ,   0   ,   0   ;
       0   ,   0   ,  200  ,   0   ,   0   ,   0   ;
       0   ,   0   ,   0   , -45   ,   0   ,   0   ;
       0   ,   0   ,   0   ,  45   ,   0   ,   0   ;
       0   ,   0   ,   0   ,   0   , -30   ,   0   ;
       0   ,   0   ,   0   ,   0   ,  30   ,   0   ;
       0   ,   0   ,   0   ,   0   ,   0   , -35   ;
       0   ,   0   ,   0   ,   0   ,   0   ,  35   ];
```

Programa naudojama trims geometrinėms konfigūracijoms patikrinti (kampų dydžiai, nurodomi pagal 2.1 skyriuje pateiktą 13 pav., indeksas *b* atitinka pagrindo kampus, *a* - platformos):

- 3-3 konfigūracija: $\alpha_b = \alpha_a = 120°, \beta_b = 2°, \beta_a = 1°, \gamma = 60°$.
- 3-6 konfigūracija: $\alpha_b = \alpha_a = 120°, \beta_b = 30°, \beta_a = 1°, \gamma = 60°$.
- 6-6 konfigūracija $\alpha_b = \alpha_a = 120°, \beta_b = 30°, \beta_a = 30°, \gamma = 60°$.

Atlikusi skaičiavimus, programa manipuliatoriaus vykdiklių postūmių (ilgių) rezultatus išspausdina į .txt failą. Visoms trims konfigūracijoms pirmosios tikrinamos vykdiklio eigos netinka. Visos likusios eigos iš sekos yra tinkamos reikalingiems parametrams užtikrinti. Gauti pirmojo iš dviejų bandymų, duomenys:

- tikrinamas vykdiklis su eiga: 600 mm;
- minimalus kojos ilgis: 600 mm;
- maksimalus kojos ilgis: 1200 mm.

2 lentelė

**Pirmosios paieškos rezultatai**

| 3-3 konfigūracija | | | | | | 3-6 konfigūracija | | | | | | 6-6 konfigūracija | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Platformos aukštis: | | | | | 322 | Platformos aukštis: | | | | | 707 | Platformos aukštis: | | | | | 806 |
| Vykdiklių ilgiai | | | | | | Vykdiklių ilgiai | | | | | | Vykdiklių ilgiai | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1103 | 1103 | 702 | 953 | 953 | 702 | 1073 | 1073 | 853 | 860 | 860 | 853 | 1023 | 1023 | 934 | 836 | 836 | 934 |
| 726 | 726 | 1119 | 915 | 915 | 1119 | 771 | 771 | 1009 | 1003 | 1003 | 1009 | 836 | 836 | 934 | 1023 | 1023 | 934 |
| 1055 | 795 | 1025 | 1136 | 674 | 833 | 938 | 930 | 1054 | 1057 | 792 | 797 | 879 | 986 | 1036 | 986 | 879 | 820 |
| 795 | 1055 | 833 | 674 | 1136 | 1025 | 930 | 938 | 797 | 792 | 1057 | 1054 | 986 | 879 | 820 | 879 | 986 | 1036 |
| 849 | 849 | 849 | 849 | 849 | 849 | 753 | 753 | 753 | 753 | 753 | 753 | 726 | 726 | 726 | 726 | 726 | 726 |
| 989 | 989 | 989 | 989 | 989 | 989 | 1064 | 1064 | 1064 | 1064 | 1064 | 1064 | 1083 | 1083 | 1083 | 1083 | 1083 | 1083 |
| 538 | 1309 | 538 | 1309 | 538 | 1309 | 840 | 1236 | 840 | 1236 | 840 | 1236 | 1078 | 1078 | 1078 | 1078 | 1078 | 1078 |
| 1309 | 538 | 1309 | 538 | 1309 | 538 | 1236 | 840 | 1236 | 840 | 1236 | 840 | 1078 | 1078 | 1078 | 1078 | 1078 | 1078 |
| 891 | 891 | 824 | 1043 | 1043 | 824 | 814 | 814 | 777 | 1172 | 1172 | 777 | 715 | 715 | 900 | 1162 | 1162 | 900 |
| 996 | 996 | 931 | 838 | 838 | 931 | 1049 | 1049 | 1008 | 725 | 725 | 1008 | 1162 | 1162 | 900 | 715 | 715 | 900 |
| 792 | 1003 | 1082 | 904 | 900 | 885 | 697 | 1149 | 1194 | 906 | 896 | 758 | 766 | 1069 | 1258 | 1069 | 766 | 687 |
| 1003 | 792 | 885 | 900 | 904 | 1082 | 1149 | 697 | 758 | 896 | 906 | 1194 | 1069 | 766 | 687 | 766 | 1069 | 1258 |

2 lentelėje ryškiau pažymėtos vertės, kurios netelpa į reikiamą ilgių intervalą, tačiau matyti jog tikrinant 6-6 konfigūraciją, netiko tik pora vykdiklių ilgių verčių. Todėl optimalių kojų ilgių paieška kartojama, susiaurinant į intervalą [600; 800], žingsniu kas 50 mm. Suformuota tokia ilgių seka: „`eiga = [600, 650, 700, 750, 800]`“;

Skaičiavimai pakartojami, gavus rezultatus matyti, jog jau antruoju bandymu gauti rezultatai rodo, kad 3-6 konfigūracijos kojų ilgiai reikiamose padėtyse telpa į reikiamą intervalą (3 lentelė):

- tikrinamas vykdiklis su eiga: 650 mm;
- minimalus kojos ilgis: 650 mm;
- maksimalus kojos ilgis: 1300 mm.

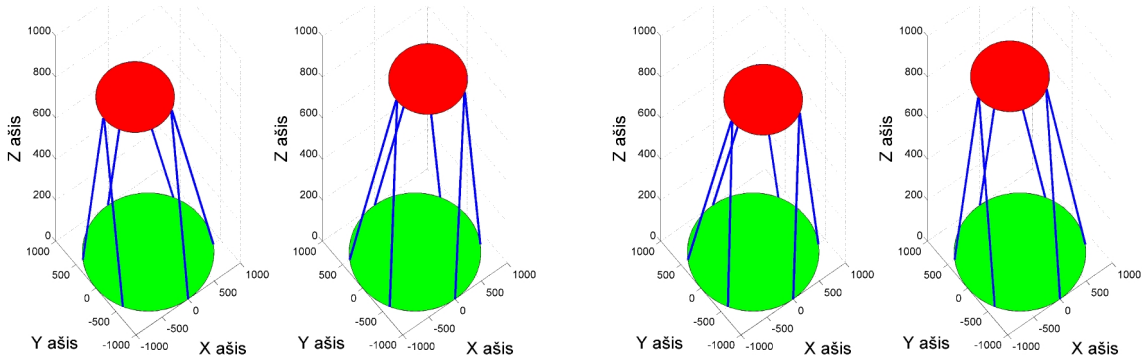
3 lentelė

**Antrosios paieškos rezultatai**

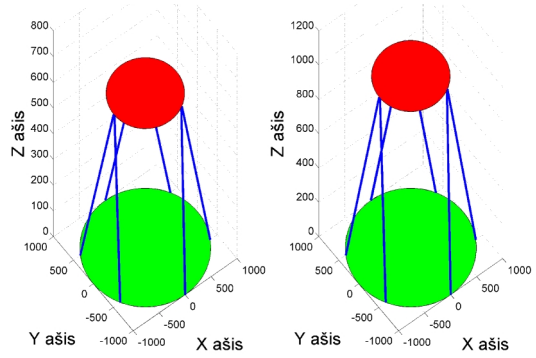
| 3-3 konfigūracija | | | | | | 3-6 konfigūracija | | | | | | 6-6 konfigūracija | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Platformos aukštis: | | | | | 494 | Platformos aukštis: | | | | | 800 | Platformos aukštis: | | | | | 889 |
| Vykdiklių ilgiai | | | | | | Vykdiklių ilgiai | | | | | | Vykdiklių ilgiai | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1165 | 1165 | 796 | 1024 | 1024 | 796 | 1136 | 1136 | 931 | 938 | 938 | 931 | 1089 | 1089 | 1007 | 916 | 916 | 1007 |
| 817 | 817 | 1180 | 989 | 989 | 1180 | 857 | 857 | 1076 | 1071 | 1071 | 1076 | 916 | 916 | 1007 | 1089 | 1089 | 1007 |
| 1119 | 879 | 1092 | 1197 | 771 | 913 | 1010 | 1003 | 1118 | 1122 | 877 | 881 | 956 | 1055 | 1101 | 1055 | 956 | 902 |
| 879 | 1119 | 913 | 771 | 1197 | 1092 | 1003 | 1010 | 881 | 877 | 1122 | 1118 | 1055 | 956 | 902 | 956 | 1055 | 1101 |
| 890 | 890 | 890 | 890 | 890 | 890 | 819 | 819 | 819 | 819 | 819 | 819 | 797 | 797 | 797 | 797 | 797 | 797 |
| 1090 | 1090 | 1090 | 1090 | 1090 | 1090 | 1145 | 1145 | 1145 | 1145 | 1145 | 1145 | 1160 | 1160 | 1160 | 1160 | 1160 | 1160 |
| 656 | 1362 | 656 | 1362 | 656 | 1362 | 920 | 1292 | 920 | 1292 | 920 | 1292 | 1141 | 1141 | 1141 | 1141 | 1141 | 1141 |
| 1362 | 656 | 1362 | 656 | 1362 | 656 | 1292 | 920 | 1292 | 920 | 1292 | 920 | 1141 | 1141 | 1141 | 1141 | 1141 | 1141 |
| 938 | 938 | 878 | 1154 | 1154 | 878 | 880 | 880 | 847 | 1253 | 1253 | 847 | 780 | 780 | 975 | 1238 | 1238 | 975 |
| 1089 | 1089 | 1028 | 860 | 860 | 1028 | 1127 | 1127 | 1088 | 781 | 781 | 1088 | 1238 | 1238 | 975 | 780 | 780 | 975 |
| 816 | 1117 | 1190 | 980 | 974 | 906 | 756 | 1231 | 1274 | 981 | 971 | 812 | 836 | 1145 | 1334 | 1145 | 836 | 745 |
| 1117 | 816 | 906 | 974 | 980 | 1190 | 1231 | 756 | 812 | 971 | 981 | 1274 | 1145 | 836 | 745 | 836 | 1145 | 1334 |

Iš gautų rezultatų (3 lentelė) galima spręsti, jog optimalus (trumpiausias) kojų ilgis reikalingas manipuliatoriui pasiekti reikiamas pozicijas yra 650 mm, o optimali geometrinė konfigūracija (konstrukcija) yra 3-6. Taip pat žemesnis platformos aukštis reiškia, jog judančios platformos masės centras bus žemiau, nei kitų bandytų geometrinių konfigūracijų, o tai reiškia, jog rastoji konstrukcija bus stabilesnė, nei likusios. Rezultatai pavaizduojami grafiškai:

- linijiniai poslinkiai:
  - *x* ašies kryptimi (25 pav. a)
  - *y* ašies kryptimi (25 pav. b)
  - z ašies kryptimi (25 pav. c)
- Ašiniai posūkiai
  - $\psi$ – apie *z* ašį (26 pav. a)
  - $\theta$ – apie *y* ašį (26 pav. b)
  - $\varphi$ – apie *x* ašį (26 pav. c)

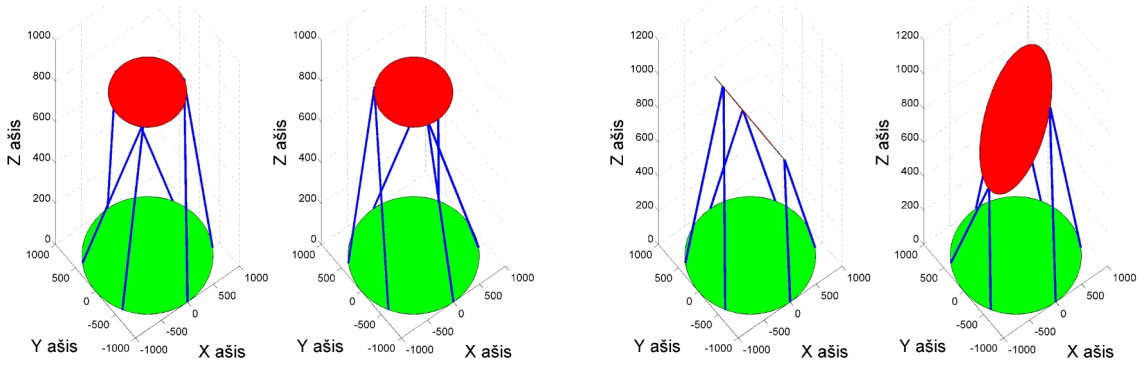



a) b)

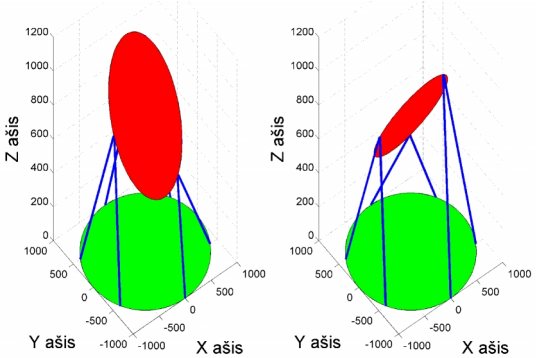



c)

25 pav. Modelio linijiniai poslinkiai




a) b)

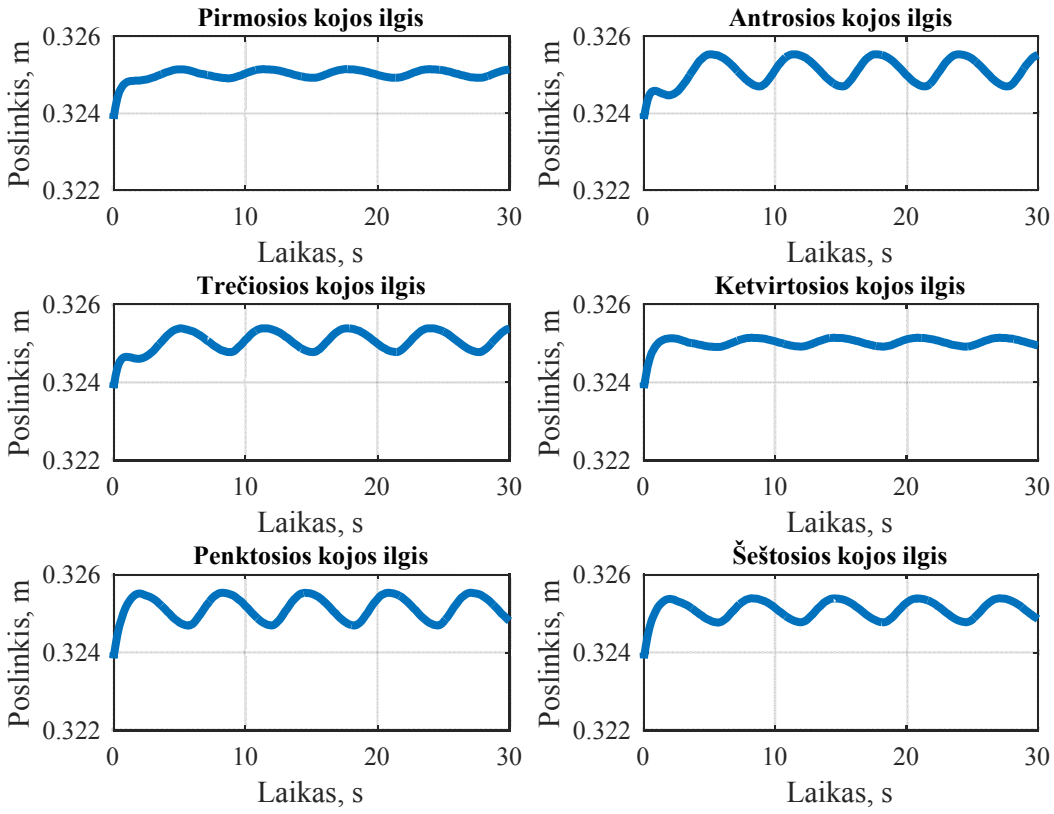



c)

26 pav.Modelio ašiniai posūkiai.

### 3.1 Stiuarto platformos vykdiklio modeliavimo rezultatai

Sudarytas Stiuarto platformos vykdiklio modelis išbandomas, jo valdymo sistemai užduodant du priešingų fazių sinusinius signalus į *x* ir *y* ašis atitinkamai. Tokia platformos valdymo komanda atitinka apvalią roboto centrinio įrankio taško trajektoriją.




27 pav.Vykdiklių ilgių kitimas pagal apvalią trajektoriją *x* ir *y* plokštumoje

Gautuose grafikuose (27 pav.) aiškiai matyti, jog pradiniu laiko momentu platformą veikia sunkio jėga, todėl ši pasislenka žemyn, bet grįžtamojo ryšio sistema, ją atstato į reikiamą padėtį. Tačiau čia modeliuojant nenaudotas joks reguliatoriaus, todėl vykdiklio dinaminis atsakas gali būti sustiprintas naudojant PID reguliatorių.

### 3.2 Stiuarto platformos vykdiklio reguliatoriaus parametrų parinkimas

Norint gauti vykdiklių reakciją į šuolinį įėjimo signalą, manipuliatoriaus valdymo sistemai užduodamas nedidelis pokytis teigiamos *z* ašies kryptimi, tokiu būdu gaunamas vykdiklių atsakas į šuolį visų bus vienodas, todėl tiriamas bus tik vieno vykdiklio atsakas į šuolį. Reakcijos kreivės aproksimavimas atliekamas dviem būdais:

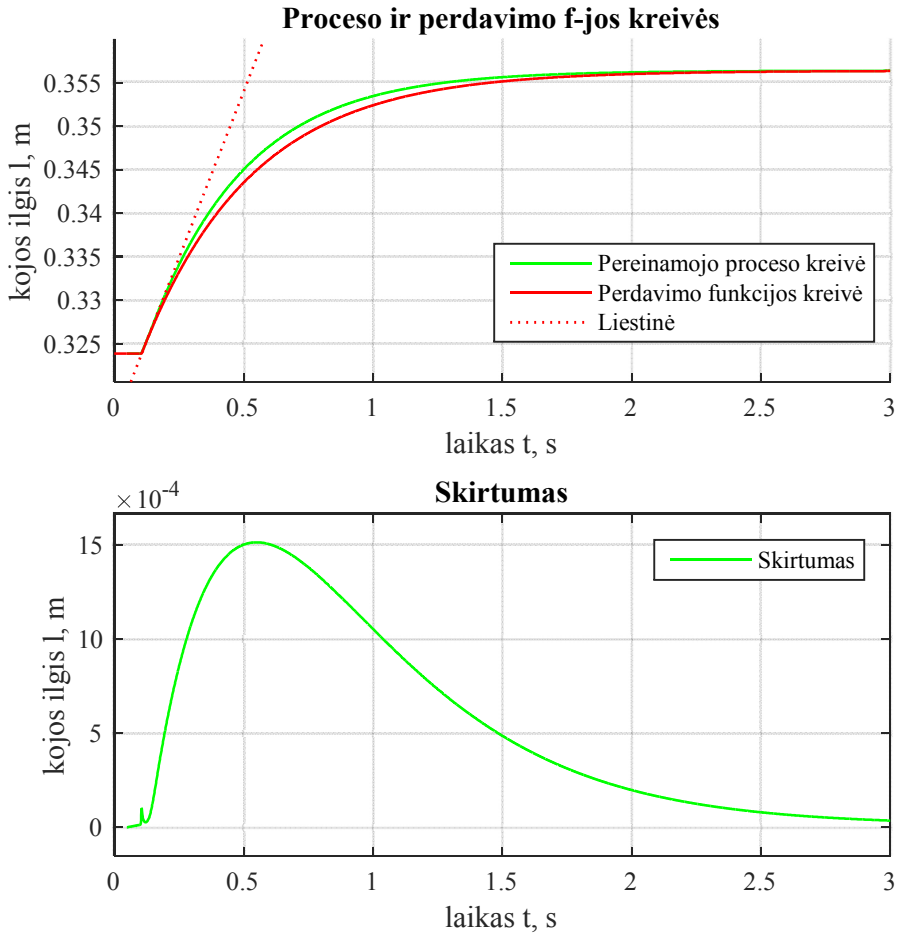
- pirmasis būdas – pereinamojo proceso reakcijos kreivės perlinkio taške (kai $v = max$) nubrėžiant kreivės liestinę;

- antrasis būdas – nubrėžiant atstojamąją tiesę per du reakcijos kreivės taškuos, esančius 28,3% ir 63,2% nusistovėjusios vertės.

Naudojant pirmąjį būdą, reakcijos kreivės funkcija identifikuojama pagal jos liestinę kreivės lūžio taške. MATLAB programa atlikus skaičiavimus, gauti rezultatai:

- vėlinimas $\tau_{pr}$ = 0,0068 s;
- laiko pastovioji $T_0$ = 0,4252 s;
- stiprinimo koeficientas $k_a$ = 0,0325.

Gautieji parametrai panaudoti perdavimo funkcijai sudaryti, sudarytoji perdavimo funkcija naudojama modeliavimui su tais pačiais parametrais, kaip ir pirminė pereinamojo proceso kreivė, gauti rezultatai pavaizduoti (28 pav.).




28 pav.Reakcijos kreivės aproksimavimas (pirmasis būdas)

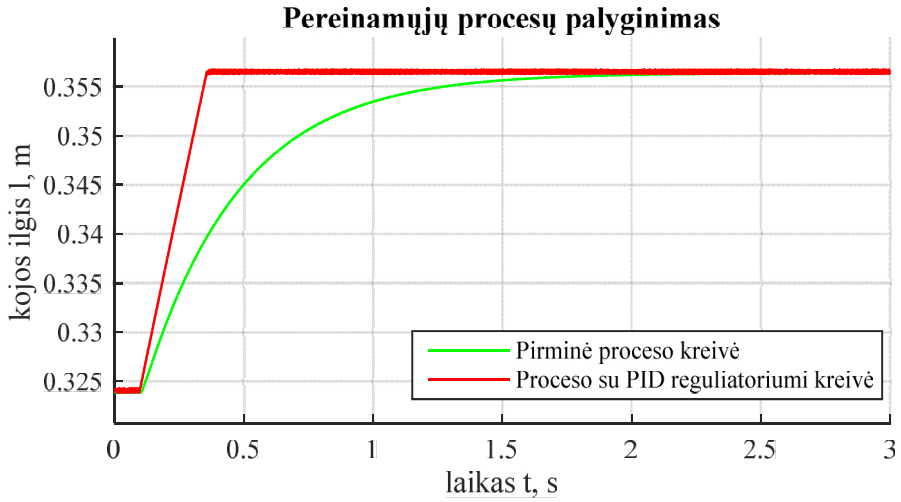
Kaip matyti iš grafikų (28 pav.), perdavimo funkcijos kreivė nuo pirminės reakcijos kreivės skiriasi labai nedaug, didžiausias skirtumas tarp kreivių – 0,43%. Įsitikinus, kad perdavimo funkcijos kreivė pakankamai atitinka pereinamojo proceso funkciją, gauti parametrai naudojami apskaičiuoti PID reguliatoriaus koeficientus. Rezultatai pateikti 4 lentelėje.

4 lentelė

**PID reguliatoriaus koeficientai**

| **Metodas** | $K_p$ | $K_i$ | $K_d$ |
|---|---|---|---|
| Ziegler ir Nichols | 2295,8 | 0,0137 | 0,0034 |
| Chien, Hrodes ir Reswick | 1817,5 | 0,0096 | 0,0032 |
| Derinimas pagal 3 dinaminius p. | 2294,8 | 0,0137 | 0,0034 |
| Cohen ir Coon | 2551,1 | 0,0181 | 0,0025 |

Iš gautųjų koeficientų matyti, kad jie visi gana panašūs, todėl modeliavimas bus atliekama tik su pirmąją iš gautųjų PID reguliatoriaus koeficientų porų – naudosime Ziegler ir Nichols metodu gautuosius koeficientus.



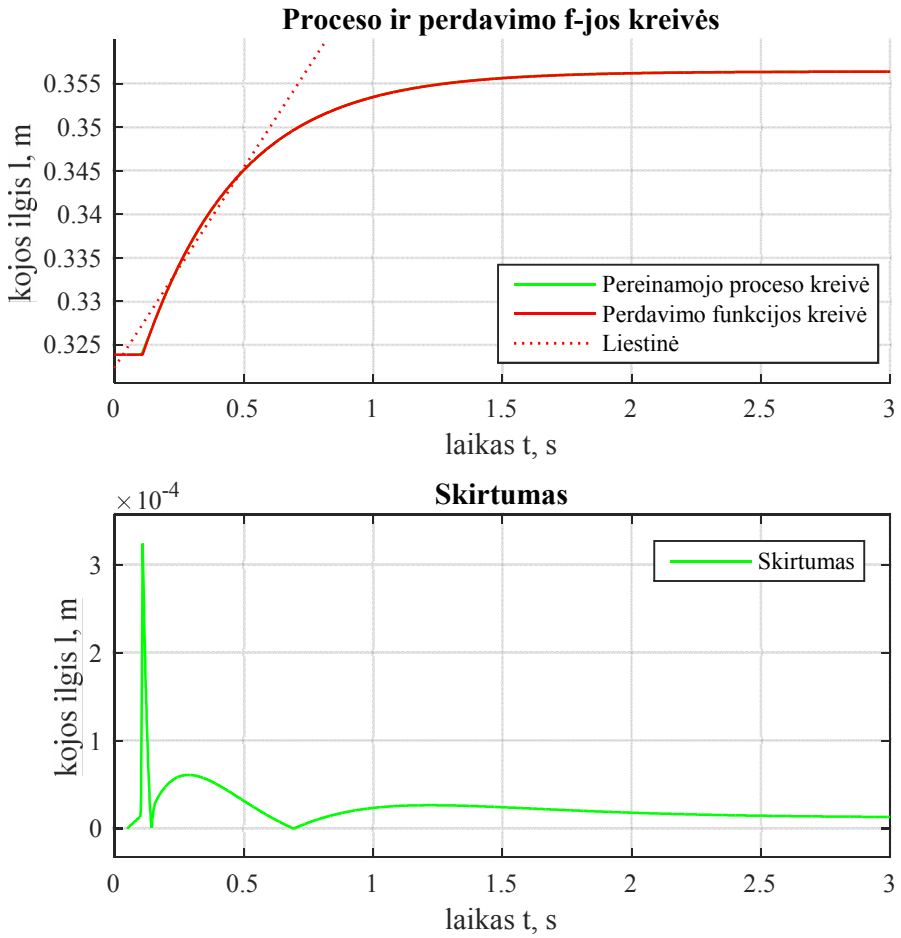

29 pav.Vykdiklio reguliatoriaus pirmasis bandymas

Iš modeliavimo rezultatų matyti, kad reakcijos kreivė, naudojant PID reguliatorių tapo aštresnė (29 pav.) ir atsirado nepageidaujami švytavimai.

Toliau bandomas reakcijos kreivės aproksimavimas, naudojant atstojamąją tiesę, nubrėžtą per du proceso kreivės taškus. Atlikus skaičiavimus MATLAB programa, gauti tokie parametrai:

- vėlinimas $\tau_{pr} = 0{,}0105$ s;
- laiko pastovioji $T_0 = 0{,}3705$ s;
- stiprinimo koeficientas $k_a = 0{,}0325$.

Gautieji parametrai panaudoti perdavimo funkcijai sudaryti. Sudarytoji perdavimo funkcija naudojama modeliavimui su tais pačiais parametrais, kaip ir pirminė pereinamojo proceso kreivė, gauti rezultatai pavaizduoti (30 pav.)




30 pav.Reakcijos kreivės aproksimavimas (antrasis būdas)

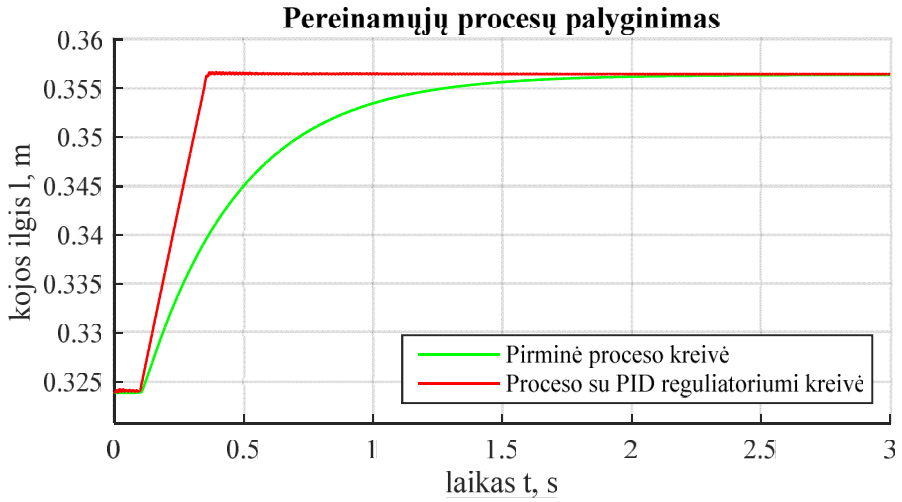
Kaip matyti iš grafikų (30 pav.), perdavimo funkcijos kreivė ir pirminė reakcijos kreivė praktiškai nesiskiria, didžiausias skirtumas tarp kreivių – 0,09%. Įsitikinus, kad perdavimo funkcijos kreivė pakankamai atitinka pereinamojo proceso funkciją, nustatyti perdavimo funkcijos parametrai naudojami apskaičiuoti PID reguliatoriaus koeficientams. Rezultatai pateikti 5 lentelėje.

5 lentelė

**PID reguliatoriaus koeficientai**

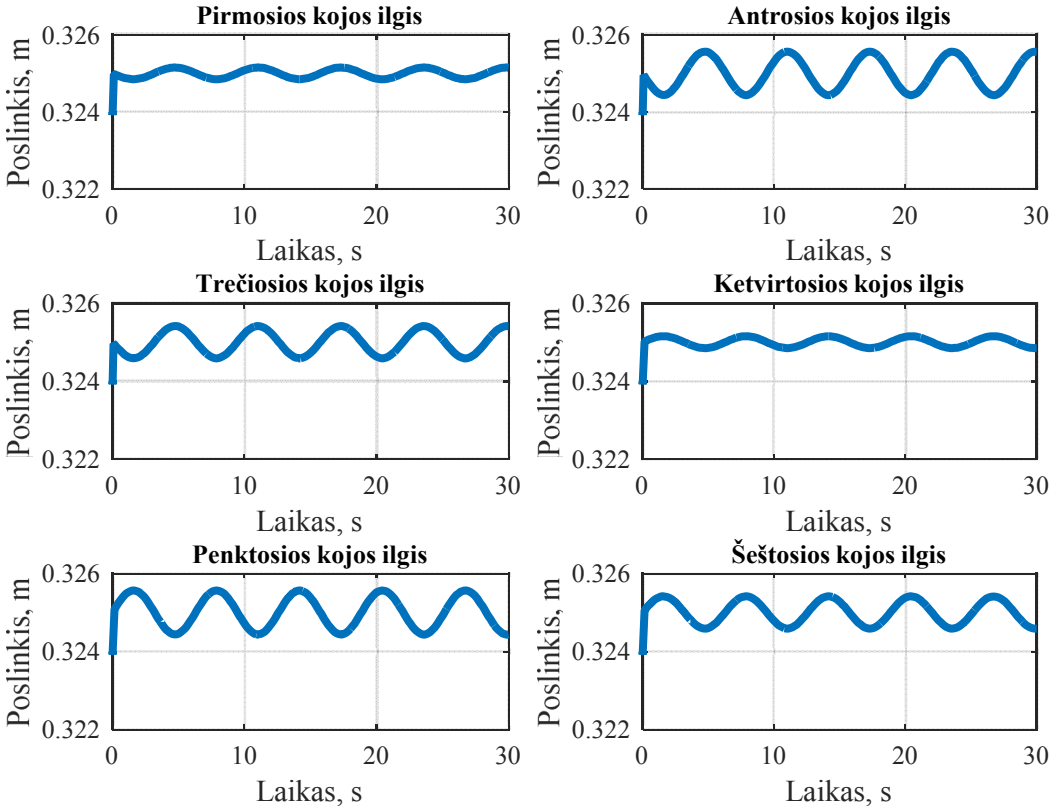
| **Metodas** | $K_p$ | $K_i$ | $K_d$ |
|---|---|---|---|
| Ziegler ir Nichols | 395,5847 | 0,0210 | 0,0053 |
| Chien, Hrodes ir Reswick | 313,1712 | 0,0147 | 0,0049 |
| Derinimas pagal 3 dinaminius p. | 1301,8 | 0,0210 | 0,0053 |
| Cohen ir Coon | 1450,5 | 0,0276 | 0,0038 |

Lyginant reguliatoriaus koeficientus, apskaičiuotus naudojant pirmąjį reakcijos kreivės aproksimavimo būdą (4 lentelė), ir gautuosius koeficientus 5 lentelėje, naudojant antrąjį būdą, galima aiškiai matyti, kad daugiausia skiriasi proporcinis reguliatoriaus koeficientas.




31 pav. Vykdiklio reguliatoriaus antrasis bandymas

Modeliavimo rezultatų kreivės pateiktos 31 pav. Iš jų aiškiai matyti, kad lyginant su pirmojo reguliatoriaus bandymo rezultatais (29 pav.), pastaruoju atveju gauti žymiai mažesni svyravimai (triukšmai), pereinamajam procesui pasiekus nusistovėjusią vertę. Todėl galima teigti, kad pastarasis metodas yra šiuo atveju tinkamesnis. Gautieji reguliatoriaus koeficientai panaudojami modeliuojant visos platformos darbą, kojų ilgių grafikai pateikti žemiau:




32 pav. Platformos judesys su PID reguliatoriais

Vykdiklio su sureguliuotu PID reguliatoriumi atsako į šuolinį pokytį valdymo signalo įėjime, gautieji dažniniai parametrai yra:

- vėlinimas $\tau_{pr}$ = 0,0280 s;
- laiko pastovioji $T_0$ = 0,1350 s;

Palyginus juos su pirminės reakcijos kreivės dinaminiais parametrais, galima daryti išvadas, kad reakcijos vėlinimas padidėjo 0.0175 sekundės, tačiau atsako greitis sutrumpėjo 2,7 karto. Padidėjęs reakcijos greitis leidžia gauti tolygesnį, nevėluojantį vykdiklių judesį sekant užduotą trajektoriją (32 pav.).

### 3.3 Skyriaus išvados

Atlikus sudarytojo Stiuarto platformos modeliavimą, padarytos tokios išvados:

- Surasto Stiuarto platformos vykdiklio ilgio leistini ribiniai judesiai telpa į reikiamus darbinės erdvės reikalavimus ir patenka į standartinių vykdiklių ilgių intervalą, todėl jį galima laikyti optimaliu.
- Modeliuojant su nustatytais optimaliais vykdiklių ilgiais ir judinant platformą $x$ ir $y$ plokštumoje, stebėta vykdiklių reakcija, kuri be reguliatoriaus yra nepakankamai greita. Todėl reikia šiai sistemai parinkti reguliatorių.
- Atlikus PID reguliatoriaus koeficientų parinkimo tyrimą, nustatyta, kad reakcijos kreivė, naudojant atstojamąją tiesę einančią per du reakcijos kreivės taškus, aproksimuojama tiksliau ir, apskaičiavus reguliatoriaus parametrus pagal Chien, Hrodes ir Reswick metodą, 2,7 karto padidinama Stiuarto platformos greitaveika.

# IŠVADOS

1. Atlikus literatūros analizę, nustatyta, kad lygiagrečios kinematinės konstrukcijos robotai, tarp jų ir Stiuarto platforma, pasižymi nesudėtingu atvirkštiniu kinematinių padėčių uždaviniu, leidžiančiu iš erdvinės padėties užduoties apskaičiuoti reikiamus vykdiklių ilgius ir taip valdyti mechanizmą.
2. MATLAB Simulink programinėje terpėje, naudojantis SimMechanics bibliotekomis, sudarytas Stiuarto platformos matematinis modelis.
3. Atliktus optimalios Stiuarto platformos vykdiklio eigos tyrimą, nustatyta, kad reikalingiems ribiniams erdviniams platformos judesiams atlikti tinkamas 650 mm eigos linijinis vykdiklis.
4. Modeliuojant Stiuarto platformą, jai parinktas PID reguliatorius, kuris padidina platformos greitaveiką 2,7 karto.

# LITERATŪROS SĄRAŠAS


1. **STEWART D.** A platform with six degrees of freedom. Proceedings of the Institute of Mechanical Engineering. Vol. 180, Part 1, No. 5, 1965, p. 371-386.
2. **MERLET J.** P. Parallel Robots (Second Edition). France, Sophia-Antipolis. INRIA, 2006. e ISBN 978-1-4020-4133-4.
3. **INDRAWANTO, SANTOSO A.** Design and Control of the Stewart Platform Robot. Third Asia International Conference on Modeling & Simulation. Mechanical Engineering Department, Faculty of Mechanical and Aerospace Engineering, Institute of Technology Bandung, 2009, p. 475-480.
4. **YANG C., HE J., HAN J., LIU X.** Real-time state estimation for spatial six-degree-of-freedom linearly actuated parallel robots. Mechatronics.19. Harbin, Hp. 1026-1033.
5. **YANG C., HUANG H. JIANG H., PETER O., HAN J.** PD control with gravity compensation for hydraulic 6-DOF parallel manipulator. Mechanism and Machine Theory 45. Harbin, Heilongjjiang, China. Elsevier, 2010, p. 666-677.
6. **SAID L., LATIFA B.** Modeling and Control of Mechanical Systems in Simulink of Matlab. Applications of MATLAB in Science and Engineering. InTech, 2011, p. 317-334.
7. **ZUBIZARRETA A., CABANES I., MARCOS M., PINTO C.** A redundant dynamic model of parallel robots for model-based control. Robotica volume 31. Cambridge University Press, 2012, p. 203-216.
8. SU **Y. X., DUAN B. Y., ZHENG C. H.** Nonlinear PID control of a six-DOF parallel manipulator. IEE Proc. Control Theory Appl. Vol. 151, No. 1. Xi‘an, China. IEE, 2003, p. 95-102
9. **MENG Q., ZHANG T., HE J., SONG J., CHEN X.** Improved model-based control of a six-degree-of-freedom Stewart platform driven by permanent magnet synchronous motors. Industrial Robot: An International Journal 39/1. Emerald Group Publishing Limited, 2012, p. 47-56.
10. **BINGUL Z., KARAHAN O.** Dynamic Modeling and Simulation of Stewart Platform. Serial and Parallel Robot Manipulators – Kinematics, Dynamics, Control and Optimization. InTech, 2012, p. 19-42.
11. **JIN Y., CHANAL H., PACCOT F.** Parallel Robots. Handbook of Manufacturing Engineering and Technology. Springer-Verlag London, 2015, p. 2091-2168. e ISBN 978-1-4471-4670-4
12. **CHEN R.** Applied robotics and parallel kinematics – Modeling and development of a Stewart – Gough platform. Master of Science Thesis, Stockholm, Sweden, 2010. Prieiga per

internetą: <http://www.diva-portal.org/smash/get/diva2:460243/FULLTEXT01.pdf>. [žiūrėta 2014 gruodžio 20 d.].

13. NASA Docking System. Prieiga per internetą: <http://en.wikipedia.org/wiki/NASA_Docking_System> [žiūrėta 2015 vasario 10 d.].
14. **MADSEN L. A., KRISTENSEN G. S.** Design of Stewart Platform for Wave Compensation. Masters Thesis. Aalborg Universitat, 2012. Prieiga per internetą: < http://projekter.aau.dk/projekter/files/63502229/EMSD415a_Final.pdf>. [žiūrėta 2014 vasario 12 d.].
15. RoboCrane. Prieiga per internetą: < http://www.cs.cmu.edu/~Xavier/robocrane.html> [žiūrėta 2015 vasario 12 d.].
16. **KŐVÁRI A.** Real-Time Modeling of an Electro-hydraulic Servo System // Computational Intelligence in Engineering, SCI 313. Springer-Verlag, Berlin, Heidelberg, 2010, p. 301-311.
17. **SOHL G. A., BOBROW E. J.** Experiments and Simulations on the Nonlinear Control of a Hydraulic Servosystem // Control Systems Technology, IEEE Transactions on (Volume: 7, Issue: 2 ). Dept. of Mech. & Aerosp. Eng., California Univ., Irvine, CA, USA, 1999, p. 178-187.
18. **RYDBERG K. E.** Hydraulic servo systems. Linköpings universitet, IEI // Fluid and Mechanical Systems. Prieiga per internetą: < https://www.iei.liu.se/ flumes/tmhp51/filearchive/coursematerial/1.105708/HydServoSystems_ part1.pdf > [žiūrėta 2015-03-14].
19. **LEVIŠAUSKAS D.** Automatinio reguliavimo sistemų derinimas. Vilniaus pedagoginio universiteto leidykla, Vilnius, 2008, p. 119. e-ISBN 978-609-02-0376-7.

# PRIEDAI

## Stiuarto platformos parametrų inicializavimo programos kodas

```
clear
clc

if(~isdeployed)
   cd(fileparts(which(mfilename)));
   addpath('FUNKCIJOS')
end

% load('init_param')
% Jei init_param.mat failas neegzistuoja reikia paleisti "const_init.m"
% Arba nuimti komentarą nuo sekančios eilutės:
run('const_init')

%Absoliuti koordinačių sistema
O  = struct('x', [1 0 0], 'y', [0 1 0], 'z', [0 0 1], 'o', [1 0 0; 0 1 0; 0 0 1]);

%Pagrindo - b ir platformos - a atraminiu taškų, bei universaliųjų grandžių
%ašių struktūros.
b  = struct('taskas', [0 0 0], 'asis1', [0 0 0], 'asis2', [0 0 0]);
a  = struct('taskas', [0 0 0], 'asis1', [0 0 0], 'asis2', [0 0 0]);
a_0 = struct('taskas', [0 0 0]);

% Kojos mechaninių kūnų koordinačių sistemos
L  = struct('vekt', [0 0 0], 'ilgis', 0, 'e', [0 0 0]);

% Cilindro korpusas
cil = struct('O', [0 0 0], 'R', eye(3), 'P', [0 0 0]);
% Kotas
kot = struct('O', [0 0 0], 'R', eye(3), 'P', [0 0 0]);

[a_0,b] = tasku_init(b, a, alfa_b, alfa_a, r_b, r_a);

[h,a] = aukstis_init(b,a_0,l); %L_0 jeigu pradine padetis ne 0

L = L_init(a,b);

[a,b] = asiu_init(a,b,L,O);


[cil, kot] = koord_init(b, L);

%Kojų masės ir inercijos momentai
[cil_m, cil_I] = inertiaCylinder(tankis, 0.75*L(1).ilgis, d_isorinis, d_vidinis);
[kot_m, kot_I] = inertiaCylinder(tankis, 0.75*L(1).ilgis, d_vidinis, 0);

% Platformos ir pagrindo masės ir inercijos momentai
[pltf_m, pltf_I] = inertiaCylinder(tankis, pltf_storis, r_a, 0);
% [pagr_m, pagr_I] = inertiaCylinder(tankis, pagr_storis, r_b, 0);

a_s = zeros(6,3);
b_s = zeros(6,3);

% for i = 1:6
%     a_s(i,:) = a_0(i).taskas;
%     b_s(i,:) = b(i).taskas;
% end
```

```
f_path = fileparts(which(mfilename));
drive = strsplit(f_path, '\');
clearvars f_path
drive= drive(1);
PID_path = [drive 'Analizavimas\PID_kof.mat'];
PID_path = strjoin(PID_path, '\');
solver_path = [drive 'solver_param.mat'];
solver_path = strjoin(solver_path, '\');

load(solver_path)
load(PID_path)

% Kp = abs(Kp);
% Ki = 1;
% Kd = 0;

clearvars f_path drive PID_path solver_path


% load_system('model_SP.slx')

% sp_plot(b,a)
% sim('model_SP')

STEP_TIME = 1e-4;
SIM_TIME = 2;

n=1;

% apskaičiuojuamas vidurio aukštis
[h0,adummy] = aukstis_init(b,a_0,L_0); %L_0 jeigu pradine padetis ne 0
dh = h0-h;

clearvars adummy h0
```

2 Priedas

## Funkcija platformos atraminių taškų vektorių apskaičiavimui

```
function [a0,b] = tasku_init(b, a, alfa_b, alfa_a, r_b, r_a)
%TASKU GENERATORIUS
b_t = b;
a_t = a;

for i = 1:3,
  kampas_minus_alfa_b = (2*pi/3)* (i-1) - alfa_b;
  kampas_plius_alfa_b = (2*pi/3)* (i-1) + alfa_b;
  b_t(2*i-1).taskas = r_b* [cos(kampas_minus_alfa_b), sin(kampas_minus_alfa_b), 0.0];
  b_t(2*i).taskas = r_b* [cos(kampas_plius_alfa_b), sin(kampas_plius_alfa_b), 0.0];

  kampas_minus_alfa_a = (2*pi/3)* (i-1) - alfa_a + pi/3;
  kampas_plius_alfa_a = (2*pi/3)* (i-1) + alfa_a + pi/3;
  a_t(2*i-1).taskas = r_a* [cos(kampas_minus_alfa_a), sin(kampas_minus_alfa_a), 0];
  a_t(2*i).taskas = r_a* [cos(kampas_plius_alfa_a), sin(kampas_plius_alfa_a), 0];
end

x = a_t;
temp = x(6).taskas;
for i = 6:-1:2
x(i).taskas = x(i-1).taskas;
end
x(1).taskas = temp;

a0 = x;
b = b_t;
end
```

3 Priedas

## Funkcija platformos aukščiui ir platformos atraminių taškų vektoriams apskaičiuoti

```
function [ h,a ] = aukstis_init( b,a_0,l )
%Fukcija platformos aukščiui rasti
a_t = struct([]);
q_v = a_0(1).taskas - b(1).taskas;
h = sqrt(l^2-norm(q_v)^2);
for i = 1:6
   a_t(i).taskas = a_0(i).taskas + [0 0 h];
end
a = a_t;
end
```

4 Priedas

## Funkcija universaliųjų sąnarų ašių parametrams apskaičiuoti

```
function [ a_r,b_r ] = asiu_init( a,b,L,O )
% Sąnarų ašių parametrų inicializacija
a_p = struct([]);
b_p = struct([]);
for i = 1:6
    a_p(i).taskas = a(i).taskas;
    b_p(i).taskas = b(i).taskas;
    % Pirmosios ašies ortas
    % Pagrindo:
    b_p(i).asis1 = cross(L(i).e, O.z);
    b_p(i).asis1 = b_p(i).asis1/norm(b_p(i).asis1);
    % Platformos:
    a_p(i).asis1 = b_p(i).asis1;

    % Antrosios ašies ortas
    % Pagrindo:
    b_p(i).asis2 = - cross(b_p(i).asis1, L(i).e);
    b_p(i).asis2 = b_p(i).asis2/norm(b_p(i).asis2);
    % Platformos:
    a_p(i).asis2 = b_p(i).asis2;
end
a_r = a_p;
b_r = b_p;
end
```

5 Priedas

## Funkcija vykdiklių (kojų) parametrams apskaičiuoti

```
function y = L_init( a, b )
%Funkcija apskaičiuoja kojų parametrus
% L = struct('vekt', [0 0 0], 'ilgis', 0, '0', [0 0 0]);
x = struct([]);
for i = 1:6,
    % Apskaičiuojamas kojos vektorius
    x(i).vekt = a(i).taskas - b(i).taskas;
    % Apskaičiuojamas kojos ilgis
    x(i).ilgis = norm(x(i).vekt);
    % Apskaičiuojamas kojos ortas
    x(i).e  = x(i).vekt / x(i).ilgis;
end
y = x;
end
```

## Funkcija vykdiklio dalių parametrams ir koordinačių ašims apskaičiuoti

```
function [ cil,kot ] = koord_init( b,L )
%Funckija judančių kūnų koordinačių sistemoms apskaičiuoti
% L = struct('vekt', [0 0 0], 'ilgis', 0, 'o', [0 0 0]);
% b = struct('taskas', [0 0 0], 'asis1', [0 0 0], 'asis2', [0 0 0]);
c = struct([]);
k = struct([]);
for i = 1:6
    vekt = L(i).vekt;
    % Cilindro:
    c(i).O = b(i).taskas + (3/8) * vekt;
    c(i).R = [b(i).asis1', b(i).asis2', L(i).e'];
    c(i).P = b(i).taskas + (3/4) * vekt;
    % Koto:
    k(i).O = b(i).taskas + (5/8) * vekt;
    k(i).R = [b(i).asis1', b(i).asis2', L(i).e'];
    k(i).P = b(i).taskas + (1/4) * vekt;
    clear vekt
end
cil = c;
kot = k;
end
```

## Funkcija vykdiklio reakcijos parametrams pagal liestinę lūžio taške rasti

```
function [ tau0, T0, b, k ] = get_tau(duom, grafikas)
%Laiko pastoviosios skaičiavimas
l = duom.l;
u = duom.u;
v = duom.v;
a = duom.a;
t = duom.t;

% Žingsnio laiko radimas
for i = 1:length(t)
   if u(i) ~= u(i+1)
      t_0 = t(i+1);
      y_0 = l(i);
      break
   end
end

[k,i] = max(v);       % Proceso kreivės perlinkio taškas

li = l(i);
ti = t(i);

liest=li+k*(t-ti);

y1 = liest(i);
y2 = liest(end);
t1 = t(i);
t2 = t(end);
b = (y2*t1-y1*t2)/(t1-t2);
k = (y1-b)/t1;

% tau0 = ((y_0-b)/k);

tau0 =(y_0-li+k*ti)/k - t_0;    % Vėlinimas
T0 = ((l(end)-b)/k) - t_0 - tau0;
% lnusist = l(end);           % Nusistovėjusi reikšmė
% d = li/lnusist;
% tau0 = ti-(1-d)*T0*log(1/(1-d));



if grafikas == 'y'
      %--------------------------------------------------------------------------

      %GRAFIKAI

      %--------------------------------------------------------------------------
      % Numatytasis ašių šriftas.
      set(0,'DefaultAxesFontName', 'Times New Roman')
      set(0,'DefaultAxesFontSize', 10)

      % Numatytasis teksto šriftas.
      set(0,'DefaultTextFontname', 'Times New Roman')
      set(0,'DefaultTextFontSize', 10)

      figure('name','Grafikai', 'position', [100, 400, 500, 250])
```

```
        title('Padėtis')
        %Grafikai
        hold on

        plot(t,l, 'LineWidth', 1, 'Color', 'Red')
        plot(t,liest, 'LineStyle',':','Color', 'b', 'LineWidth', 2)
        legend('Reakcijos kreivė', 'Liestinė')
        hold off

        %Ašys ir tinklelis
        ylim([l(1)-0.01*l(1) l(end)+0.01*l(end)])
        xlabel('laikas t, s')
        ylabel('kojos ilgis l, m')
        grid on

end
end
```

8 Priedas

**Funkcija vykdiklio reakcijos parametrams pagal du taškus reakcijos kreivėje rasti**

```
function [ tau0, T0, b, k ] = get_2p( duom, grafikas )
%Parametrų apskaičiavimas pagal du taskus
x = duom.l;
u = duom.u;
t = duom.t;
d = x(end) - x(1);

% Žingsnio laiko radimas

for i = 1:length(t)-1
   if u(i) ~= u(i+1)
      t_stp = t(i+1);
      break
   end
end

i1 = [];
i2 = [];
e = 1e-6;
k = e;
y1 = 0.283 * d + x(1)
y2 = 0.632 * d + x(1)
while length(i1) < 1
   i1 = find(x > y1-k & x < y1+k);
   k = k + e;
end
k = e;
while length(i2) < 1
   i2 = find(x > y2-k & x < y2+k);
   k = k + e;
end
t1 = t(i1)
t2 = t(i2)

b = (y2*t1-y1*t2)/(t1-t2);
k = (y1-b)/t1;

T0 = 3/2*(t2-t1);
tau0 = t2 - T0-t_stp;
```

```
t_lin = linspace(0,t(end));
y_lin = k*t_lin+b;

if grafikas == 'y'
    %--------------------------------------------------------------------------

    %GRAFIKAI

    %--------------------------------------------------------------------------
    % Numatytasis ašių šriftas.
    set(0,'DefaultAxesFontName', 'Times New Roman')
    set(0,'DefaultAxesFontSize', 10)

    % Numatytasis teksto šriftas.
    set(0,'DefaultTextFontname', 'Times New Roman')
    set(0,'DefaultTextFontSize', 10)

    figure('name','Grafikai', 'position', [100, 100, 600, 300])

    title('Tiesė per 2 taškus')
    %Grafikai
    hold on
%     plot(t,u, 'LineWidth', 1, 'Color', 'Green')
    plot(t,x, 'LineWidth', 1, 'Color', 'Blue')
    plot(t_lin,y_lin, 'LineWidth', 1, 'Color', 'Red')
    line([t1 t2],[y1 y2], 'LineWidth', 3, 'Color', 'Black')
    legend('Reakcijos kreivė', 'Liestinė')
    hold off

    %Ašys ir tinklelis
    ylim([x(1)-0.01*x(1) x(end)+0.01*x(1)])
    xlabel('laikas t, s')
    ylabel('kojos ilgis l, m')
    grid on
end
end
```

## Funkcija 3-6 konfiguracijos optimaliems kojų ilgiams rasti

```
clear
clc
%--------------------------------------------------------------------------
%
%                   *3-6 KONFIGURACIJA*
%
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%                       Veiksmai su failais
%--------------------------------------------------------------------------

c = clock;
d =  sprintf('SP 3-6 %i-%i-%i %i_%i.txt', c(1), c(2), c(3), c(4), c(5));
f = fopen( d, 'wt' );
fprintf( f, '-------- Raportas --------\n\n');

%--------------------------------------------------------------------------
%                       Tikrinamų parametrų sekos
%--------------------------------------------------------------------------

%Tikrinamu cilindrų eigų seka:
stroke = [600, 650, 700, 750, 800];
%Atliekamų veiksmų parametrų matrica
%   [   x  ,  y   ,  z   ,  psi , teta ,  fi  ]
V = [ -250 ,  0   ,  0   ,  0   ,  0   ,  0   ;
      250 ,  0   ,  0   ,  0   ,  0   ,  0   ;
      0   , -250 ,  0   ,  0   ,  0   ,  0   ;
      0   ,  250 ,  0   ,  0   ,  0   ,  0   ;
      0   ,  0   , -200 ,  0   ,  0   ,  0   ;
      0   ,  0   ,  200 ,  0   ,  0   ,  0   ;
      0   ,  0   ,  0   , -45  ,  0   ,  0   ;
      0   ,  0   ,  0   ,  45  ,  0   ,  0   ;
      0   ,  0   ,  0   ,  0   , -30  ,  0   ;
      0   ,  0   ,  0   ,  0   ,  30  ,  0   ;
      0   ,  0   ,  0   ,  0   ,  0   , -35  ;
      0   ,  0   ,  0   ,  0   ,  0   ,  35  ];



%--------------------------------------------------------------------------
%                       Pradiniai duomenys
%--------------------------------------------------------------------------

%Platformos ir bazės spinduliai, metrais
r_base = 1000;
fprintf( f, 'Bazės apskritimo spindulys: \n%i\n', r_base);
r_top = 600;
fprintf( f, 'Platformos apskritimo spindulys: \n%i\n\n', r_top);



%Centriniai taskai
O_base = [0 0 0];
%Poslinkiu vektorius
```

```
T = [0 0 0];

%------------------------Geometrinė konfiguracija-------------------------
%Platformos ir bazės kojų taškų atlenkimo nuo 120° atskaitos kampas, deg.
alpha_base_deg = 30;
alpha_top_deg = 1;

%Isdestymo kampai
pos_angle = degtorad(120);
top_rot_angle = degtorad(60);

alpha_base = degtorad(alpha_base_deg);
alpha_top = degtorad(alpha_top_deg);

pos_base = zeros(6, 3);
pos_top = pos_base;

for i=1:3
   angle_minus_b = (pos_angle)* (i-1) - alpha_base;
   angle_plus_b = (pos_angle)* (i-1) + alpha_base;
   pos_base(2*i-1,:) = r_base* [cos(angle_minus_b), sin(angle_minus_b), 0.0];
   pos_base(2*i,:) = r_base* [cos(angle_plus_b), sin(angle_plus_b), 0.0];

   angle_minus_t = (pos_angle)* (i-1) - alpha_top + top_rot_angle;
   angle_plus_t = (pos_angle)* (i-1) + alpha_top + top_rot_angle;
   pos_top(2*i-1,:) = (r_top* [cos(angle_minus_t), sin(angle_minus_t), 0]);
   pos_top(2*i,:) = (r_top* [cos(angle_plus_t), sin(angle_plus_t), 0]);

end

% platformos kojų lietimosi tasku seka pastumiama į priekį per vieną
% elementą taip kad bazės ir platformos i-toji koja su baze ir platforma
% liestusi i-tuosiuose taskuose
pos_top = [pos_top(6,:); pos_top(1:5,:)];

%Atstumo nuo taško bi iki taško pi projekcija į xy plokštumą.
q_xy = norm(pos_base(1,:) - pos_top(1,:));

%--------------------------------------------------------------------------
%                         SKAICIAVIMAI
%--------------------------------------------------------------------------
%Pradedamas cilindrų eigų tikrinimas
for i = 1 : numel(stroke)
   fprintf( f, 'Tikrinamas cilindras su eiga:\t%i\n', stroke(i));
   %----------------------- Kojų/Cilindrų ilgiai -------------------------

   %Tikrinamas abstraktus teorinis cilindro modelis, laikant kad cilindro
   %ilgis lygus koto ilgiui, bei eigai, o "namų" koto padėtis yra pusė
   %jo eigos
   %Cilindro ilgis:
   Lc = stroke(i);
   %i - tojo cilindro ilgis "namų" pozicijoje:
   Li = Lc + (stroke(i)/2);
   %Minimalus kojos ilgis laikomas cilindro ilgiu:
   Lmin = Lc;
   fprintf( f, 'Minimalus kojos ilgis:\t%i\n', Lmin);
   %Maksimalus kojos ilgis laikomas pilnai išstumto koto, bei cilindro
```

```
   %ilgių suma
   Lmax = Lc + stroke(i);
   fprintf( f, 'Maksimalus kojos ilgis:\t%i\n', Lmax);

   %------------------ Pradiniai platformos parametrai -------------------

   %Ieskomas platformos aukstis "namų" pozicijoje.
   %Platformos aukšis:
   h = sqrt((Li)^2 - (q_xy)^2);
   fprintf( f, 'Platformos aukštis:\t%i\n', uint32(h));
   %Platformos taškų poslinkio vektorius
   T(3) = T(3)+h;
   %Platformos koordinačių sistemos pradžia
   O_top = O_base + T;

   eil_count = size(V);
   leg = zeros(1,6);

   for j = 1:eil_count(1)
      p_top = next_p(pos_top,T,V(j,:));
      for k = 1:6
         leg(k) = abs(sqrt((p_top(k,1)-pos_base(k,1))^2+(p_top(k,2)-pos_base(k,2))^2+(p_top(k,3)-
pos_base(k,3))^2));
         if or( leg(k) < Lmin , leg(k)> Lmax)
            fprintf( f, '%iX(!), ', uint32(leg(k)));
            %break;
         else
            fprintf( f, '%i, ', uint32(leg(k)));
         end
      end
      fprintf( f, '\n');
   end

   fprintf( f, '\n');

   %--------------------- Parametru inicializacija ------------------------
   %Isvalomos cikliškai naudojamų kintamųjų sekos
   T = zeros (1, 3);
   fprintf( f, '\n\n');
end
fclose('all');



text = sprintf('Įveskite cilindrų eigų sekos indeksą, pagal kurį bus nubraižytas grafikas\n');
in = input(text);
%----------------------- Kojų/Cilindrų ilgiai -------------------------
%Cilindro ilgis:
Lc = stroke(in);
%i - tojo cilindro ilgis "namų" pozicijoje:
Li = Lc + (stroke(in)/2);
h = sqrt((Li)^2 - (q_xy)^2);
%Platformos taškų poslinkio vektorius
T(3) = T(3)+h;
sp_graf(O_base,O_top,r_base,r_top,pos_base,pos_top,T,V)
```