VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

Tomas LUNECKAS

ŠEŠIAKOJO ROBOTO JUDĖJIMO NELYGIU PAVIRŠIUMI TYRIMAS

DAKTARO DISERTACIJA

TECHNOLOGIJOS MOKSLAI,
ELEKTROS IR ELEKTRONIKOS INŽINERIJA (01T)

VGTU
LEIDYKLA
TECHNIKA
Vilnius 2013

Disertacija rengta 2009–2013 metais Vilniaus Gedimino technikos universitete.

Mokslinis vadovas

doc. dr. Dainius UDRIS (Vilniaus Gedimino technikos universitetas, technologijos mokslai, elektros ir elektronikos inžinerija – 01T).

VGTU leidyklos TECHNIKA 2193-M mokslo literatūros knyga
http://leidykla.vgtu.lt

ISBN 978-609-457-584-6

© VGTU leidykla TECHNIKA, 2013
© Tomas Luneckas, 2013
tomas.luneckas@vgtu.lt

VILNIUS GEDIMINAS TECHNICAL UNIVERSITY

Tomas LUNECKAS

ANALYSIS OF HEXAPOD ROBOT MOVEMENT OVER ROUGH TERRAIN

DOCTORAL DISSERTATION

TECHNOLOGICAL SCIENCES,
ELECTRICAL AND ELECTRONIC ENGINEERING (01T)

VGTU
LEIDYKLA
TECHNIKA
Vilnius 2013

Doctoral dissertation was prepared at Vilnius Gediminas Technical University in 2009–2013.

Scientific Supervisor

Assoc Prof Dr Dainius UDRIS (Vilnius Gediminas Technical University, Technological Sciences, Electrical and Electronic Engineering – 01T).

Reziumė

Disertacijoje nagrinėjamas šešiakojo roboto judėjimas nelygiu paviršiumi ir paviršiaus savybių atpažinimas ir vertinimas pagal roboto pėdų padėtis. Sudarytas ir ištirtas algoritmas, kuris leidžia atpažinti ir įvertinti paviršiaus savybes pagal roboto pėdų koordinates.

Disertaciją sudaro įvadas, trys skyriai, rezultatų apibendrinimas, naudotos literatūros ir autoriaus publikacijų disertacijos tema sąrašas.

Įvadiniame skyriuje aptariama tiriamoji problema, darbo aktualumas, aprašomas tyrimų objektas, formuluojamas darbo tikslas bei uždaviniai, aprašoma tyrimų metodika, darbo mokslinis naujumas, darbo rezultatų praktinė reikšmė, ginamieji teiginiai. Įvado pabaigoje pristatomos disertacijos tema autoriaus paskelbtos publikacijos ir pranešimai konferencijose bei disertacijos struktūra.

Pirmajame skyriuje pateikiama paviršiaus klasifikavimo apžvalga ir išskiriami kriterijai pagal kuriuos galimas paviršiaus savybių klasifikavimas. Taip pat aptariami pagrindiniai paviršiaus savybių atpažinimo metodai. Apžvelgiami skaitiniai paviršiaus savybių vertinimo metodai. Skyriaus pabaigoje formuluojamos išvados ir tikslinami disertacijos uždaviniai.

Antrajame skyriuje pateiktas šešiakojo roboto kinematinio ir imitacinio modelio sudarymas. Aprašoma sudaryta supaprastinta roboto valdymo sistema bei valdančioji programa. Sudaromas paviršiaus savybių atpažinimo ir vertinimo algoritmas bei sprendimų priėmimo sistema.

Trečiajame skyriuje pateikiami standartinės nuokrypos verčių nustatymo rezultatai imituojant roboto ėjimą nežinomais paviršiais. Tiriamas fizinio roboto modelio judėjimas lygiu ir nelygiu paviršiumi ir aptariamos nuokrypos bei jų kompensavimo ir paviršiaus nelygumo vertinimo rezultatai. Apibendrinami eisenos parinkimo algoritmo imitavimo rezultatai.

Disertacijos tema yra atspausdinti 4 moksliniai straipsniai recenzuojamuose mokslo žurnaluose, 5 – kituose leidiniuose: vienas – mokslo žurnale, įtrauktame į *Thomson ISI Proceedings* sąrašą; trys – Lietuvos mokslo žurnaluose; penki – kituose recenzuojamuose mokslo leidiniuose. Disertacijos tema perskaityta 12 pranešimų mokslinėse konferencijose.

Abstract

Dissertation analyses hexapod robot locomotion over rough terrain and terrain feature identification without using any specialized sensors. An algorithm for identifying and evaluating terrain features using robot‘s feet coordinates is introduced.

The thesis consists of introduction, three main chapters, conclusions, bibliography, list of publication.

Introduction reveals problem and importance of the work and the object of research as well as describes the purpose and tasks of thesis, research methodology, scientific novelty, practical significance of obtained results and defended statements. Finally a list of author’s publications and conference presentations is given as well as dissertation structure.

First chapter introduces to terrain classification and features for terrain classification are described. Also main methods for detecting terrain features are introduced. At the end of the chapter, conclusions and tasks for dissertation are formulated.

Second chapter presents robot’s kinematic and imitation model creation. Designed simplified robot control system and control program are described. Also terrain feature identification and evaluation method and decision making system are developed.

In the third chapter standard deviation value estimation by simulating robot locomotion on different terrain is presented. Physical robot’s locomotion on flat terrain using adaptive gait is investigated and feet coordinate deviation compensation is discussed. Also robot’s locomotion over rough terrain is investigated and results for terrain roughness identification are presented. Furthermore gait selection algorithm simulation results are discussed.

Results of the work were published 4 scientific articles in reviewed scientific periodical publications, 5 – in other editions: one – published in journal quoted in *Thomson ISI Proceedings* databases, three – in periodic Lithuanian journals, five – in referred international conference journals. 12 presentations were made in conferences on the subject.

Simboliai ir santrumpos

Simboliai

B – pagalbinė menamoji linija, jungianti šlaunies pradžią ir blauzdos pabaigą;

$\mathbf{B}_i$ – homogeninė transformacijų matrica;

$\mathbf{d}_{XYZ}$ – roboto kūno koordinačių sistemos centro koordinačių matrica;

h – žingsnio aukštis;

i, j, k – kojų indeksai;

l – žingsnio ilgis;

l_1 – šlaunies ilgis;

l_2 – blauzdos ilgis;

l_3 – atstumas tarp klubo ir šlaunies ašių;

l_4 – linijos B projekcijos į xy plokštumą ilgis;

L_1 – roboto kūno ilgis;

L_2 – roboto kūno plotis;

p_1, p_2, p_3 – roboto valdymo kriterijų prioritetai;

r_1, r_2, r_3, r_4 – eisenų reitingų suminės vertės;

$\mathbf{R}_{XYZ}$ – posūkio apie visas tris ašis matrica;

$\mathbf{R}_X, \mathbf{R}_Y, \mathbf{R}_Z$ – posūkio matricos apie *X*, *Y* ir *Z* ašis;

T – žingsnio periodas;

$\mathbf{T}_i$ – kojų pagrindų poslinkio matrica;

$\mathbf{T}_{XYZ}$ – homogeninės posūkio apie kūno koordinačių ašis transformacijų matrica;

w_1, w_2, w_3 – roboto valdymo kriterijų svoriai;

x_i, y_i, z_i – tikrosios pėdų koordinatės, naudojamos atvirkštinės kinematikos skaičiavimuose;

x_{fi}, y_{fi}, z_{fi} – pėdų koordinatės;

$x_{B_i}, y_{B_i}, z_{B_i}$ – kojų pagrindų koordinatės;

$x_{B_i\_offset}, y_{B_i\_offset}, z_{B_i\_offset}$ – kojos pagrindo koordinačių sistemos poslinkiai;

X_0, Y_0, Z_0 – roboto kūno koordinačių sistemos centro koordinatės;

$X_{B_i}, Y_{B_i}, Z_{B_i}$ – kojų pagrindų koordinatės kūno koordinačių sistemoje;

α – roboto kūno posūkis apie *x* ašį;

α_0 – roboto kūno pokrypio korekcijos kampas *x* ašimi;

α_x – roboto kūno pokrypis *x* ašimi išmatuotas akcelerometru;

α_y – roboto kūno pokrypis *y* ašimi išmatuotas akcelerometru;

α_1, α_2 – pirmosios ir antrosios plokštumų pokrypio kampai *x* ašimi;

β – roboto kūno posūkis apie *y* ašį;

β_0 – roboto kūno pokrypio korekcijos kampas *y* ašimi;

β_1, β_2 – pirmosios ir antrosios plokštumų pokrypio kampai *y* ašimi;

γ – roboto kūno posūkis apie *z* ašį;

ε – roboto ėjimo kryptis;

θ_1 – klubo pavaros kampas;

θ_2 – šlaunies pavaros kampas;

θ_3 – alkūnės pavaros kampas;

σ – standartinė nuokrypa;

φ_i – kojos fazė.

Santrumpos

J-LF – kairės priekinės kojos jutiklis;
J-LH – kairės galinės kojos jutiklis;
J-LM – kairės vidurinės kojos jutiklis;
J-RF – dešinės priekinės kojos jutiklis;
J-RH – dešinės galinės kojos jutiklis;
J-RM – dešinės vidurinės kojos jutiklis;
LF – kairė priekinė koja;
LH – kairė galinė koja;
LM – kairė vidurinė koja;
RF – dešinė priekinė koja;
RH – dešinė galinė koja;
RM – dešinė vidurinė koja.

Turinys

Contents

Įvadas

Problemos formulavimas

Šiuolaikiniai robotai vis dažniau ir plačiau pradedami taikyti nežinomoje dinamiškoje aplinkoje. Atvirose lauko sąlygose robotams tenka prisitaikyti prie besikeičiančių išorinių veiksnių. Vienas iš svarbiausių aplinkos veiksnių, kuriuos turi įveikti mobilūs robotai, yra paviršiaus nelygumai. Važiuojantys robotai sunkiai įveikia kliūtis, kurių aukštis yra pusė jų rato skersmens ir negali įveikti kliūčių didesnių už šį aukštį.

Šio trūkumo neturi žingsniuojantys robotai, kurie gali judėti labai nelygiu paviršiumi (Görner, Stelzer 2013) bei peržengti gilius griovius. Populiariausi yra dvikojai, keturkojai ir šešiakojai robotai. Kiekviena iš šių robotų rūšių pasižymi tam tikrais privalumais ir trūkumais. Šešiakojai robotai daug dėmesio sulaukia dėl savo statinio stabilumo (Manoiu-Olaru *et al.* 2011), tai yra jiems nereikia papildomos įrangos stabilumui palaikyti ir todėl yra tinkami judėjimui labai nelygiu paviršiumi. Kadangi daugeliu atveju šešiakojai robotai turi 18 laisvės laipsnių, jų valdymas tampa sudėtingas (Haynes *et al.* 2012; Sorin, Mircea 2012).

Viena iš svarbiausių problemų, robotams judant atviroje aplinkoje, yra nežinomo nelygaus paviršiaus atpažinimas ir vertinimas. Norint sėkmingai judėti paviršiumi, pirmiausia, aplinka, kurioje dirba robotas, turi būti atpažinta ir kie-

kybiškai įvertinta. Tik tada roboto valdymo sistema gali priimti teisingą sprendimą, kaip robotui elgtis esamoje situacijoje (Kajita, Tani 1996).

Šiandieniniai sėkmingiausi robotai, galintys judėti nelygiu paviršiumi, yra valdomi vieno ar kelių kompiuterių, kurie veikia pagal sudėtingus algoritmus. Todėl naujų, paprastesnių ir greičiau veikiančių, aplinkos atpažinimo ir vertimo metodų kūrimas yra labai svarbi robotų inžinerijos problema, kurios išsprendimas leis plačiau pritaikyti žingsniuojančius robotus ir suteiks jiems autonomiškumą.

Darbo aktualumas

Paviršiaus savybių atpažinimas ir vertinimas yra viena iš svarbiausių robotų inžinerijos problemų. Kurios išsprendimas nulems gerokai platesnį mobilių robotų taikymą. Dėl šios priežasties reikia tobulinti esamus ir kuri naujus paviršiaus savybių atpažinimo ir vertinimo metodus.

Literatūroje galima rasti paviršiaus savybių suskirstymo į kategorijas aprašymų. Toks suskirstymas leidžia lengviau įvertinti paviršiaus savybes ir praeinamumą. Tačiau nėra aiškiai įvardijamas paviršiaus savybių fizinių dydžių pagrįstumas. Todėl viena iš problemų yra konkrečių paviršiaus savybių fizinių dydžių apibrėžimas ir santykio su roboto matmenimis nustatymas.

Viena iš didžiausių neišspręstų problemų yra paviršiaus savybių atpažinimas ir vertinimas. Šiuo metu, paviršiaus savybių atpažinimui, plačiausiai naudojamos vaizdo kamerų ir lazerinių atstumo jutiklių sistemos. Norint su šiomis sistemomis išgauti duomenis apie paviršių, reikalingi sudėtingi algoritmai. Todėl tarp duomenų gavimo, jų apdorojimo ir sprendimo priėmimo gali susidaryti laiko delsos, kurios įtakos roboto veikimą realiu laiku ir robotas negalės reaguoti į staigiai besikeičiančias aplinkos sąlygas. Be to vaizdinės aplinkos atpažinimo sistemos yra priklausomos nuo matomo vaizdo ryškumo (aiškumo) ir gali prastai veikti (arba išvis neveikti) esant blogam matomumui. Dėl šių priežasčių yra svarbu sukurti metodus, kurie leistų atpažinti paviršiaus savybes naudojant ne vaizdinę informaciją bei paprastus algoritmus, kuriuos galėtų vykdyti įterptinis roboto valdiklis.

Atpažinus paviršiaus savybes, svarbu juos įvertinti ir teisingai interpretuoti. Šiuo metu, aplinkos objektai arba paviršius skirstomas į robotui praeinamą, sunkiai praeinamą arba visiškai nepraeinamą. Stengiamasi išskirti tik tokias kliūtis, kurios trukdo robotui judėti. Tačiau, žingsniuojantys robotai gali judėti kur kas nelygesniu paviršiumi, nei važiuojantys, todėl paviršiaus savybės turėtų būti skirstomi į gerokai daugiau kategorijų arba turi tiesiogiai įtakoti roboto judėjimą ir elgseną.

Tyrimų objektas

Darbo tyrimų objektas – šešiakojis žingsniuojantis robotas ir jo judėjimo nežinomu nelygiu paviršiumi valdymas.

Darbo tikslas

Šio darbo pagrindinis tikslas yra sudaryti paviršiaus savybių atpažinimo, vertinimo bei roboto eisenos parinkimo metodus ir ištirti šešiakojo roboto judėjimą nelygiu paviršiumi.

Darbo uždaviniai

Norint pasiekti darbo tikslą reikia spręsti šiuos uždavinius:

1. Sudaryti roboto imitacinį ir fizinį modelius bei sudaryti trajektorijų generavimo matematines išraiškas adaptyviam roboto ėjimui.
2. Ištirti roboto ėjimą lygiu paviršiumi, taikant adaptyvią eiseną ir įvertinti paviršiaus atpažinimo nuokrypas.
3. Sudaryti ir ištirti metodą, leidžiantį atpažinti paviršiaus savybes pagal esamas roboto pėdų koordinates.
4. Sudaryti ir ištirti šešiakojo žingsniuojančio roboto eisenos parinkimo pagal paviršiaus savybes algoritmą.

Tyrimų metodika

Darbe taikomi atvirkštinės kinematikos, homogeninių transformacijų, trajektorijų generavimo, statistinės analizės metodai. Roboto kinematinis modelis imituojamas MATLAB® aplinkoje, eksperimentiniai tyrimai atliekami su fiziniu roboto modeliu.

Darbo mokslinis naujumas

Rengiant disertaciją buvo gauti šie elektros ir elektronikos inžinerijos mokslui nauji rezultatai:

1. Sudarytas šešiakojo žingsniuojančio roboto valdymo algoritmas, kuriame realizuotos realiu laiku generuojamos pėdų trajektorijos, leidžiantis robotui prisitaikyti prie paviršiaus nelygumų.
2. Sudarytas naujas paviršiaus savybių atpažinimo ir vertinimo metodas, kuris leidžia įvertinti paviršiaus savybes naudojantis tik žinomomis roboto pėdų koordinatėmis ir roboto kūno pokrypiais.
3. Sudarytas naujas roboto sprendimų priėmimo algoritmas, kuris leidžia parinkti roboto eiseną, priklausomai nuo paviršiaus savybių ir iš anksto apibrėžtų reikalavimų roboto atliekamai užduočiai.

Darbo rezultatų praktinė reikšmė

Tyrimų rezultatai gali būti naudojami šešiakojų žingsniuojančių robotų valdymo sistemose paviršiaus savybių atpažinimui ir vertinimui. Tai leidžia praplėsti esamų aplinkos atpažinimo sistemų galimybes ir užtikrinti patikimesnį paviršiaus atpažinimą ir vertinimą. Sudaryta sprendimų priėmimo algoritmas leidžia parinkti eiseną priklausomai nuo paviršiaus savybių, taip užtikrinant sėkmingesnį roboto judėjimą nelygiu paviršiumi. Sudaryti šešiakojo roboto imitacinis ir fiziniai modeliai gali būti taikomi šešiakojų robotų judesių ir valdymo algoritmų tyrimui, robotui judant įvairiais paviršiais.

Ginamieji teiginiai

1. Pėdų koordinačių nuokrypų, atsirandančių taikant adaptyvią eiseną, kompensavimas valdymo sistemoje leidžia nuokrypas vidutiniškai sumažinti 65 % trikojei, 31 % keturkojei, 77 % banguojančiai ir 85 % pulsuojančiai eisenoms, lyginant su ėjimu netaikant kompensavimo.
2. Pėdų *z* koordinačių nuokrypų kompensavimas pagal roboto kūno pokrypius leidžia paviršiaus nelygumų aukščių atpažinimą pagerinti iki 2,7 karto, neapibrėžtis sumažinti iki 2,2 karto, standartinės nuokrypos neatitikimą sumažinti iki 9,25 karto, o standartinės nuokrypos neapibrėžtis sumažinti iki 1,5 karto.
3. Sudaryti ir eksperimentiškai patikrinti šešiakojo roboto imitacinis ir fizinis modeliai bei sudarytas eisenos parinkimo algoritmą leidžia šešiakojams robotams taikyti adaptyvų judėjimą nežinomu nelygiu paviršiumi ir parinkti eiseną pagal paviršiaus savybes.

Darbo rezultatų aprobavimas

Disertacijos tema yra atspausdinti 4 moksliniai straipsniai recenzuojamuose mokslo žurnaluose ir 5 – kituose leidiniuose: vienas – mokslo žurnale, įtrauktame į Thomson ISI Proceedings sąrašą; trys – mokslo žurnaluose, cituojamuose IndexCopernicus duomenų bazėje, vienas – mokslo žurnale, cituojamame Scopus duomenų bazėje, keturi – kituose recenzuojamuose mokslo leidiniuose.

Disertacijoje atliktų tyrimų rezultatai buvo paskelbti dvylikoje mokslinių konferencijų Lietuvoje ir užsienyje:

- Jaunųjų mokslininkų konferencijose *Mokslas – Lietuvos ateitis* 2010–2012 m. Vilniuje;
- Tarptautinėse konferencijose *Elektros ir valdymo technologijos* 2009–2013 m. Kaune.
- Tarptautinėse konferencijose *Elektronika* 2010, 2011 m. Kaune;
- Tarptautinėse konferencijose *Mechatronic Systems and Materials* 2009 Vilniuje ir 2010 m., Kaune.

2012 metais autorius dalyvavo *Locomorph summer school on morphology and morphosis in animals and robots* vasaros mokykloje Pietų Danijos universitete, Odense mieste.

Disertacijos struktūra

Disertaciją sudaro įvadas, trys skyriai ir rezultatų apibendrinimas.

Darbo apimtis yra 103 puslapiai, tekste panaudotos 63 numeruotos formulės, 40 paveikslų ir 8 lentelės. Rašant disertaciją buvo panaudoti 79 literatūros šaltiniai.

1

Paviršiaus savybių vertinimo metodų apžvalga

Skyriuje išskiriami pagrindiniai paviršiaus savybių klasifikavimo kriterijai. Aptariami pagrindiniai robotų inžinerijoje sutinkami paviršiaus savybių atpažinimo metodai. Apžvelgiami metodai, kurie leidžia kiekybiškai įvertinti ir apibūdinti paviršiaus savybes. Skyriaus pabaigoje pateikiamos išvados ir tikslinami darbo uždaviniai.

1.1. Paviršių klasifikavimas

Paviršiai gali pasižymėti skirtingomis savybėmis, kurios nulemia reikiamą robotų elgseną. Suklasifikavus paviršius, kurie pasižymi vienodomis ar panašiomis savybėmis, galima nustatyti kokios robotų elgsenos bus efektyviausios judant tokiu paviršiumi ir roboto valdymo sistema galės parinkti tinkamiausią elgseną (Larson *et al.* 2005; Giguere, Dudek 2009).

Paviršius galima apibūdinti pagal tris pagrindines savybes (Terrain classification 2013): paviršiaus tvirtumas, paviršiaus nelygumas ir paviršiaus pokrypis (šlaitas).

- Paviršiaus tvirtumas (Davis, Reisinger 1990) – apibūdina žemės paviršiaus klampumą, kuris priklauso nuo žemės birumo ir drėgnumo.
- Paviršiaus nelygumas – apibūdina paviršiaus kliūtis, kurios gali trukdyti judėjimui.
- Paviršiaus pokrypis – apibūdina paviršiaus plokštumos kampą su horizontu.

Visi trys kriterijai gali būti apibendrinti vienu parametru – paviršiaus praeinamumu.

Norint nusakyti koks yra paviršius, jis pagal kiekvieną savybę yra suskirstomas į tam tikras apibrėžtas kategorijas (1.1 lentelė) (Terrain classification 2013).

1.1 lentelė. Paviršiaus savybių kategorijos
Table 1.1. Terrain feature categories

Paviršiaus tvirtumas		Paviršiaus nelygumas		Šlaitas	
1	Labai geras	1	Labai lygus	1	Horizontalus
2	Geras	2	Lygus	2	Nuožulnus
3	Vidutinis	3	Truputi nelygus	3	Vidutiniškas
4	Prastas	4	Nelygus	4	Status
5	Labai prastas	5	Labai nelygus	5	Labai status

Kiekviena kategorija turi būti aprašoma fiziniais dydžiais (matmenimis) apibūdinančiais paviršiaus savybes. Paviršiaus tvirtumo kategorijos nurodo kokio kietumo ir klampumo yra paviršius. Kuo žemesnė savybės kategorija, tuo klampesnis, minkštesnis yra paviršius. Taigi, aukščiausiai kategorijai priskiriamas žvyras, kietas smėlis ir pan. Žemiausiai kategorijai – labai šlapios durpės ar tiesiog vanduo (Terrain classification 2013; Davis, Reisinger 1990). Paviršiaus būsena yra tiesiogiai siejama su drėgme ar metiniu kritulių kiekiu ir grunto birumu.

Paviršiaus nelygumas yra siejamas su kliūčių dydžiu ir jų tankiu. Kliūčių dydis (Terrain classification 2013; Davis, Reisinger 1990) pateikiamas nuo 10 cm iki >90 cm. Kokiai nelygumo kategorijai bus priskirtas paviršius priklauso nuo to kokio dydžio kliūtys ir kokiu tankiu jos yra pasiskirstę paviršiaus ploto vienete. Kuo daugiau didesnių kliūčių ploto vienete, tuo aukštesnei nelygumo kategorijai paviršius priskiriamas. Tačiau nei kliūčių aukščiai, nei jų tankis nėra siejami su roboto matmenimis, t. y. nėra pagrindžiami tokie aukščių matmenys.

Paviršiaus šlaitas yra pakankamai save paaiškinanti savybė ir jos kategorijos yra siejamos su paviršiaus nuolydžiu, pateikiamu procentais. Labai stataus šlaito kategorijai priskiriami šlaitai, kurių nuolydis yra apie 45–50 % (21,8–26,6º).

Pabrėžtina tai, kad nė viename šaltinyje nepateikiamas kategorijų fizinių dydžių pagrindimas. Nėra aišku su kokiais fiziniais dydžiais yra siejami paviršiaus savybių kriterijai.

1.1.1. Paviršiaus nelygumas

Viena iš svarbiausių paviršiaus savybių, į kurią reikia atsižvelgti, yra paviršiaus nelygumas, kuris, kaip jau buvo minėta, apibūdina kliūčių dydžius. Paviršiaus nelygumo apibūdinimui svarbu išskirti kategorijas. Vienas iš būdų yra paviršiaus nelygumą klasifikuoti pagal kliūčių tankį bei jų dydį. Kliūčių dydis gali būti nusakomas dviem kategorijomis {Maža, Didelė}. Kliūtis vienai iš šių kategorijų priskiriama atlikus erdvinio vaizdo (gauto dviem vaizdo kameromis) analizę ir suskaičiavus kiekvieną kliūtį sudarančių taškų kiekį. Pagal iš anksto apibrėžtą taškų kiekį, kliūtis yra priskiriama arba mažai, arba didelei. Taip pat, norint nusakyti bendrą paviršiaus nelygumą kliūčių tankiai yra suskirstomi į kategorijas. Mažos ir didelės kliūtys yra sugrupuojamos ir pagal tai nustatomas kliūčių tankis, kuris priskiriamas vienai iš kategorijų: {Mažai, Daug}. Bendras paviršiaus nelygumas yra išreiškiamas iš kliūčių dydžio ir jų tankio kategorijų, taikant neraiškiąją logiką, ir priskiriamas vienai iš nelygumo kategorijų: {Lygus, Nelygus, Akmenuotas}. Tačiau nėra pateikiami konkretūs kliūčių dydžio fiziniai matmenys ar tankiai. Taigi, lieka neaišku kokio dydžio kliūtys ir koks jų tankis sudaro, pavyzdžiui *Nelygų* paviršių (Seraji *et al.* 2001).

Paviršiaus nelygumas gali būti nepriskiriamas iš anksto apibrėžtoms kategorijoms (Ishigami *et al.* 2011), tačiau pateikiamas kaip standartinė paviršiaus iškilimų nuokrypa po robotu. Šis paviršiaus nelygumas yra tiesiogiai susijęs su roboto pravažiuojamumu, t. y. priklauso nuo roboto matmenų.

Vienas iš siūlomų metodų paviršiaus nelygumo apibūdinimui yra susijęs su kliūčių buvimu ar nebuvimu aplinkoje. Šis parametras yra įvardijamas kaip kliūtingumas (Ettlin, Bleuler 2006; Seraji, Bon 2002). Šis parametras apibūdina kliūčių buvimą roboto darbo aplinkoje ir yra lygus 0, jei kliūčių nėra, arba lygus 1, jei roboto darbo aplinkoje yra neįveikiamos kliūtys. Šis parametras gali būti tinkamas kliūčių tankio apibūdinimui, tačiau neteikia jokios informacijos apie kliūčių matmenis. Be to, šis parametras yra naudojamas roboto kelio planavimui, nes nusako ar numatomame kelyje yra nepereinamų (nepravažiuojamų) kliūčių, todėl jis tinkamesnis važiuojančių robotų valdymui.

1.1.2. Šlaitas ir paviršiaus tvirtumas

Mažiau nagrinėjamos paviršiaus savybės yra šlaitas arba paviršiaus gradientas ir paviršiaus tvirtumas, kuris nusako paviršiaus kietumą (minkštumą). Šios savybės negali būti paprastai aptinkami vaizdinėmis ar taktilinėmis roboto jutiklių sistemomis, lyginant su paviršiaus nelygumu. Tačiau šios paviršiaus savybės taip pat įtakoja roboto judėjimą ir kelio planavimą.

Šlaito kampą galima nustatyti iš erdvinio vaizdo (Seraji *et al.* 2001) arba įvertinant esamą roboto korpuso pokrypį (Ishigami *et al.* 2011). Pirmuoju atveju, šlaito pokrypis yra priskiriamas vienai iš kategorijų: {Lygus, Nuožulnus, Status}. Antruoju atveju, skaičiuojamos dvi paviršiaus šlaito kampo dedamosios, du pokrypio kampai. Iš šių dedamųjų išrenkamas didžiausias kampas, kuris toliau naudojamas esamam šlaitui apibūdinti. Tačiau šlaitas pagal apskaičiuotą kampą nėra priskiriamas jokiai kategorijai.

Panašiai kaip ir su paviršiaus nelygumu, nėra pateikiamos konkrečios šlaito kampo vertės, kurios leistų nusakyti koks šlaitas yra status. Analizuojamas roboto ratų praslydimas ir roboto galimybė apsiversti ant šono, priklausomai nuo šlaito kampo, tačiau taip pat nėra nurodomos tikslios kampų vertės šiems įvykiams atsitikti (Ishigami *et al.* 2011).

Paviršiaus tvirtumo įvertinimui gali būti taikoma vaizdų analizė, taikant dirbtinių neuronų tinklus (Howard, Seraji 2001) paviršiaus tekstūros palyginimui su iš anksto žinomomis. Paviršiaus tvirtumas yra tiesiogiai priklausomas nuo paviršiaus medžiagos (minkštas smėlis, nesuspaustas žvyras, presuota žemė ir pan.) ir drėgnumo. Išmokius sistemą atpažinti minėtų medžiagų tekstūrą, paviršiaus būsena priskiriama vienai iš kategorijų: {Minkšta, Vidutinė, Kieta} (Ishigami *et al.* 2011).

1.1.3. Paviršiaus praeinamumas

Norint supaprastinti paviršiaus apibūdinimą, visi minėti paviršiaus vertinimo kriterijai yra kombinuojami į vieną bendrą paviršiaus apibūdinimo parametrą – paviršiaus praeinamumą. Kaip tiksliai reiktų apibūdinti paviršiaus praeinamumą nėra visiškai aišku. Teigiama, kad paviršiaus praeinamumas turėtų atitikti roboto galimybę sėkmingai įveikti paviršiaus plotą.

Paviršiaus praeinamumas suskaidomas į tris kategorijas: {Žemas, Vidutinis, Aukštas}, kurios parenkamos pagal nagrinėjamo paviršiaus ploto nelygumą ir šlaitą. Taip pat, galimas paviršiaus praeinamumo klasifikavimas vietinėms, regioninėms ir globalioms zonoms (Seraji, Bon 2002). Praeinamumas yra suskirstomas į kategorijas: {Prastas, Žemas, Vidutinis, Aukštas}. Paviršiaus praeinamumas priskiriamas vienai iš šių kategorijų priklausomai nuo paviršiaus nelygumo, šlaito ir tvirtumo, taikant neraiškiąją logiką (1.2 ir 1.3 lentelės) (Seraji, Bon 2002; Molino *et al.* 2007).

1.2 lentelė. Neraiškiosios logikos taisyklės lokalaus praeinamumo įvertinimui (Seraji, Bon 2002)
Table 1.2. Fuzzy logic rule set for evaluating local traversability

		Atstumas iki artimiausios kliūties		
		Toli	Arti	Labi arti
Paviršiaus tvirtumas	Kietas	Aukštas	Vidutinis	Prastas
	Vidutinis	Vidutinis	Žemas	Prastas
	Mikštas	Žemas	Prastas	Prastas

1.3 lentelė. Neraiškiosios logikos taisyklės regiono praeinamumo įvertinimui (Seraji, Bon 2002)
Table 1.3. Fuzzy logic rule set for evaluating regional traversability

		Paviršiaus nelygumas			
		Lygus	Nelygus	Duobėtas	Akmenuotas
Paviršiaus šlaitas	Lygus	Aukštas	Aukštas	Vidutinis	Prastas
	Pakrypęs	Aukštas	Vidutinis	Žemas	Prastas
	Vidutinis	Vidutinis	Žemas	Žemas	Prastas
	Status	Prastas	Prastas	Prastas	Prastas

Paviršiaus praeinamumas gali būti apibūdinamas pavojingumo parametru (Stelzer *et al.* 2012). Paviršiaus regionui priskiriamas pavojaus parametras *d*, apibūdinantis to regiono praeinamumo sudėtingumą. $d = 0$ apibūdina visiškai lygų, tvirtą, plokščią paviršių. Kai $d = 1$, paviršiaus regionas laikomas beveik nepraeinamu robotui. Visiškai nepraeinamam paviršiui priskiriamos pavojaus vertės $d = \infty$. Nežinomam paviršiui priskiriamos $d = 1$ vertės.

Paviršiaus pavojaus kriterijų nulemia tokios paviršiaus savybės: šlaito statumas, didelis paviršiaus nelygumas ir aukšti slenksčiai. Jei šių kriterijų vertės viršija apibrėžtas vertes, paviršiaus regionui priskiriama nepraeinamo paviršiaus vertė (Stelzer *et al.* 2012). Svarbu pastebėti, kad šiame straipsnyje ribinės kriterijų vertės nustatomos pagal tai, kada robotas apsiverčia ar užstringa.

1.2. Paviršiaus savybių atpažinimo metodai

Nors paviršiaus apibūdinimas ir savybių klasifikavimas gali būti nesudėtingas ir gana paprastai išsprendžiamas uždavinys, daug sunkumų kyla įvertinant minėtas paviršiaus savybes. Nors tokios savybės kaip paviršiaus nelygumas ar šlaito pokrypis gali būti išmatuoti jutikliais tokiais kaip optiniai atstumo jutikliai, ultragarsiniai jutikliai, akcelerometrai, tačiau paviršiaus tvirtumą yra itin sudėtinga nustatyti – nėra patikimų jutiklių bei metodų. Nors esama įranga įgalina bent dalinai nustatyti paviršiaus savybes, tačiau jų nustatymui reikia arba atskirų specializuotų jutiklių sistemų arba yra taikomi sudėtingi algoritmai, reikalaujantys sparčių valdiklių.

Vienas iš universaliausių metodų, norint nustatyti paviršiaus savybes, yra erdvinių vaizdų analizė gautų naudojant dvi vaizdo kameras. Tokia sistema, pritaikius tinkamus metodus ir algoritmus, iš kamerų gaunamo vaizdo gali gauti reikiamą informaciją apie paviršių bei aplinką. Didžiausias tokios sistemos trūkumas yra labai sudėtingi algoritmai, bei reikiamų vaizdų analizės algoritmų nebuvimas. Šie trūkumai labiausiai ir apriboja platų erdvinių vaizdų analizės taikymą robotų valdymo sistemose (Yunde *et al.* 2003; Manduchi *et al.* 2004; Hornung *et al.* 2010).

Tačiau tokių sistemų perspektyvumas nulemia didelį susidomėjimą jų tobulinimu. Vaizdo kameros, lyginant su kitomis jutiklių sistemomis, turi keletą privalumų, tokių kaip didesnė raiška, mažesnis jautrumas kliūčių formoms bei atspindžiams (Chow, Chung 2002).

Robotams taikomos vaizdinės sistemos dažniausiai naudoja dvi kameras (Chow, Chung 2002; Yunde *et al.* 2003; Kolter *et al.* 2009), nes norint iš gauto vaizdo nustatyti gylį, atstumą iki objektų, yra reikalingi du to paties vaizdo truputi perslinkti vienas kito atžvilgiu kadrai. Nuskaityti paviršiaus vaizdą naudo-

jant kameras yra paprasta, tačiau reikalingi sudėtingi metodai bei algoritmai paviršiaus savybėms apskaičiuoti.

Tiriant vabzdžius buvo sukurtas metodas, leidžiantis analizuoti matomą vaizdą taikant optinio srauto (angl. *optic flow*) principą (Beauchemin, Barron 1995; Lewis 2002; Lengvinis *et al*. 2011). (Lewis 2002) pateikiamas optinio srauto metodo taikymas žingsniuojančiam robotui. Metodas leidžia pastebėti greičio kitimą dviem kryptimis (1.1) ir (1.2).

$$\frac{dx}{dt} \propto \int \frac{E_t\left(\nabla E \cdot [1\ 0]^T\right)}{\left\|\nabla E\right\|^2}, \tag{1.1}$$

$$\frac{dy}{dt} \propto \int \frac{E_t\left(\nabla E \cdot [1\ 0]^T\right)}{\left\|\nabla E\right\|^2}, \tag{1.2}$$

čia dx/dt yra šoninis greitis (1.1), dy/dt – išilginis greitis (1.2).

Pagrindinė optinio srauto problema yra atsirandantys srauto nukrypimai dėl roboto judesių aukštyn ir žemyn robotui einant netgi lygiu paviršiumi. Kadangi roboto judesiai yra periodiniai, galimas problemos sprendimo būdas yra numatyti ir kompensuoti žinomus optinio srauto nukrypimus. Nors optinio srauto metodas gali aptikti kliūtis ir įvertinti jų parametrus (Lewis 2002; Chow, Chung 2002), tačiau gaunamas signalas yra labai triukšmingas ir dar kelia daug problemų aptinkant nedidelius paviršiaus pokyčius. Šis metodas yra tinkamesnis lygiu keliu važiuojantiems robotams, nes tokiu atveju gaunamas vaizdas yra labai stabilus.

Kitas metodas paviršiaus savybių nustatymui taikant kameras yra ne vertinti vaizdo judesius, o jo erdvinius parametrus (Chow, Chung 2002). Norint sumažinti skaičiavimų sudėtingumą, siūlomas metodas nėra taikomas viso erdvinio vaizdo gavimui ir vertinimui. Iš erdvinio vaizdo išgaunama tik informacija apie kliūčių padėtis aplink robotą (Chow, Chung 2002). Tokiu būdu sumažėja skaičiavimų trukmė ir tai leidžia robotui greičiau reaguoti į aplinkos pokyčius. Toks informacijos supaprastinimas neleidžia įvertinti pagrindinių paviršiaus savybių, t. y. paviršiaus nelygumų, tvirtumo ar šlaitų. Ši informacija tik leistų įvertinti kliūčių tankį, ko neužtenka sėkmingam roboto judėjimui nelygiu paviršiumi.

Norint pagerinti erdvinio vaizdo informatyvumą nusakant gylį, vienas iš variantų yra naudoti tris kameras (Yunde *et al*. 2003). Erdvinio vaizdo išgavimui iš tokios kamerų sistemos algoritmas turi būti sudarytas iš tokių trijų pagrindinių dalių: 1) vaizdų transformacijos ir horizontalaus filtravimo taikant Gauso filtrą; 2) vertikalaus filtravimo taikant Gauso ir Laplaso filtrus; 3) dvigubo 1D SSAD

iteracinio algoritmo ir papildomo apdorojimo. Visas šis algoritmas gali būti atliekamas specializuotame FPGA valdiklyje (Yunde *et al.* 2003).

Galimas paviršiaus žemėlapio sudarymo metodas, kai dviem vaizdo kameromis du kartus per sekundę fiksuojamas vaizdas yra sulyginamas ir sutapatinamas su prieš tai fiksuotu vaizdu, taikant Artimiausio Iteracinio Taško metodą (Kolter *et al.* 2009). Tokiu būdu sudarinėjant paviršiaus žemėlapį, lieka nemažai nematomų paviršiaus dalių, kadangi kameros negali matyti paviršiaus už kliūčių. Tai sudaro sunkumų planuojant roboto judesius ir jo kelią.

Paprastam kliūčių aptikimui gali būti pakankamas koreliacija pagrįstas atstumo iki objektų skaičiavimo metodas (Hirschmüller *et al.* 2002). Tačiau buvo pastebėta, kad SGM (angl. *Semi-Global Matching*) metodo (Hirschmüller 2008) taikymas yra tinkamesnis, nes juo gaunami rezultatai yra tikslesni ir reikalaujantys mažiau sąnaudų. Be to, šis metodas leidžia aptikti plonus arba mažus objektus, kurie dažniausiai lieka neaptikti taikant koreliacijos metodą.

SGM yra pagrįstas vaizdų sugretinimu taškelių (angl. *pixel*) lygyje, pasitelkiant globalų vienalytiškumo apribojimą (Stelzer *et al.* 2012). Gaunama globali svorio koeficientų funkcija yra minimizuojama pagal aštuonias kelio kryptis, prasidedančias ties kadro riba. Taigi, visi aštuoni keliai kertasi kiekviename taškelyje. Jie yra kombinuojami sumuojant jų svorių koeficientus ir skirtumus, kurie minimizuoja svorių koeficientus. Svorio koeficientai yra parenkami atskirai kiekvienam taškeliui. Neatitikimai yra identifikuojami ir anuliuojami pritaikant kairės (dešinės) atitikimo patikrą, kuri atlieka abiejų kamerų paskirčių inversiją ir panaikina visus neatitikimus.

Vaizdinė odometrija yra roboto padėties ir orientacijos nustatymas dviem vaizdo kameromis fiksuojamos aplinkos atžvilgiu. Vaizdinė odometrija leidžia nustatyti judesio kryptį ir dydį iš mažiausiai dviejų kadrų. Stereo kameros leidžia apskaičiuoti visus šešis laisvės laipsnius (poslinkius ir postūmius), lyginant su pavienėmis kameromis, kurios tik įgalina nustatyti judesio kryptį, bet ne jo dydį. Vaizdinė odometrija yra nepriklausoma nuo ratų ar kojų praslydimų ir gali būti pritaikyta visiems mobiliems robotams judėti bet kokiu paviršiumi, tačiau turi būti priimta, kad aplinka yra statiška. Taip pat, kadangi kadravimo dažnis yra ganėtinai mažas, gali susidaryti dideli judesio pokyčiai tarp gretimų kadrų, ypač jei robotas sukasi, pvz. į kairę arba dešinę. Tačiau rezultatai parodė, kad vaizdinė odometrija nėra pakankama aplinkos atpažinimui ar roboto padėties nustatymui, nes robotui judant atsiranda paklaidos ir rezultatai nukrypsta. Tikslius rezultatus galima gauti vaizdinę odometriją kombinuojant su kitų jutiklių duomenimis, pvz. kojos odometrija (Stelzer *et al.* 2012; Guizilini, Ramos 2013; Germanas, Narvydas 2009).

Norint nustatyti paviršiaus nelygumų dydį ar jų tankį, svarbiausias uždavinys iš kamerų vaizdo yra nustatyti matomo vaizdo gylį, o tam vaizdo kameros nėra pritaikytos. Ši problema gali būti nesunkiai išsprendžiama aplinkos atpaži-

nimui taikant lazerinius atstumo jutiklius. Lazeriniai atstumo jutikliai aplinkos vaizdą pateikia kaip atstumų iki artimiausių kliūčių masyvą. Taikant lazerinius atstumo jutiklius atkrenta poreikis iš matomo vaizdo nustatyti atstumus iki kliūčių. Tačiau tokia sistema neleidžia nustatyti paviršiaus tvirtumo (Rekleitis *et al.* 2013; Premebida, Nunes 2013).

Jei aplinkos paviršiaus nelygumų matmenys daug didesni nei lazerinio atstumo jutiklio spindulio plotis, išmatuoto atstumo paklaidos priklauso nuo fizinių jutiklio galimybių ir yra mažos bei apytiksliai pasiskirsto pagal Gauso dėsnį (Browning *et al.* 2012). Paviršius pavaizduojamas trikampių tinklu su spindulių sekimu (angl. *ray tracing*), geba išgauti priimtiną realybės aproksimaciją. Toks metodas yra dažnai naudojamas komercinių robotų imitavimo aplinkose (Koenig, Howard 2004; Carpin *et al.* 2007; Gonzalez *et al.* 2009) ir gerai atpažįsta tokias medžiagas kaip dirva, sausas cementas, plytinė siena ir kt. Nesudėtingi spindulių sekimo būdo papildymai gali leisti sumodeliuoti itin atspindinčius (pvz. kvarcą, metalus) ar permatomus (pvz. vanduo, stiklas) paviršius. Tačiau šis metodas nėra tinkamas augalijos atpažinimui (Browning *et al.* 2012).

Lazerinio atstumo jutiklio taikymas augalijos atpažinimui yra sudėtingas dėl daugelio krypčių atspindžių, sugerties ir įvairių, skirtingai šviesą atspindinčių ar sugeriančių, reiškinių augalo ląstelėse (pvz. lapo ląstelių struktūroje (Jones, Vaughan 2010)). Be to, sudėtinga augalų paviršių geometrija sukuria papildomus daugelio krypčių atspindžius. Šis efektas pasireiškia kiekvieno paviršiaus ribose (Tuley *et al.* 2005), lazerinio atstumo jutiklio spindulys atsitrenkęs į du ar daugiau paviršių skirtinguose gyliuose gali sukurti vieną ar daugiau grįžtamųjų spindulių, kurie gali nesutapti su tikruoju atstumu iki matuojamo paviršiaus. Šis efektas vadinamas „sumaišytų taškelių atsaku (angl. *mixed-pixel returns*)“. Dėl sudėtingos struktūros ir mažų paviršių matmenų, artimų lazerinio atstumo jutiklio spindulio pločiui, augmenijos sumaišytų taškelių atsakai gali sudaryti didžiąją išmatuotų duomenų dalį ir iškraipyti tikruosius duomenis. Paprasti šio efekto kompensavimo metodai šiuo atveju nėra tinkami, kadangi jie neteisingai įvertina atsakų kampines priklausomybes.

Galimas metodas, kai lazerinio atstumo jutiklio duomenys yra integruojami į trimatį taškų „debesį“, robotui judant (Stavens, Thrun 2006). Duomenų integravimui yra reikalingos apytikrės koordinatės, kuriose buvo atliktas matavimas ir jutiklio orientacija, arba roboto padėtis. Integravimas atliekamas taikant Kalmano filtrą (Julier, Uhlmann 1997) su 100 Hz atnaujinimo dažniu. Dėl matavimų triukšmo ir duomenų perdavimo vėlinimų padėties nustatyme gali atsirasti paklaidos. Dėl to, trimačiai taškų debesys negali būti priimti neapdirbti ir matavimų triukšmo atskyrimas yra viena iš pagrindinių užduočių (Stavens, Thrun 2006). Padėties nustatymo paklaidas yra sudėtinga sumodeliuoti, eksperimentai parodė, kad paklaidos linkusios didėti laike (Thrun *et al.* 2006).

Matuojant paviršių savybes naudojant lazerinio atstumo jutiklį didžiausi netikslumai atsiranda dėl šviesos fizikinių savybių, kurios nulemia šviesos išsisklaidymą ore, lūžimą esant paviršiaus nelygumų matmenims artimiems spindulio pločiui.

Aplinkos atpažinimui robotuose neretai yra naudojama Microsoft kompanijos įrenginys Kinect™. Kinect™ yra sudarytas iš vienos RGB vaizdo kameros ir gylio jutiklio, kuris sudaytas iš infraraudonųjų spindulių lazerio ir monochromatinio CMOS jutiklio. Kadangi Kinect™ yra sudarytas iš kelių skirtingų jutiklių, jis dažniausiai taikomas roboto pozicijos nustatymui ir tuo pat metu aplinkos žemėlapio sudarymui (angl. *Simultaneous Localization And Mapping (SLAM)*). Nors SLAM yra skaitomas vienas iš esminių proveržių robotikoje, tačiau dėl duomenų nuskaitymo paklaidų ir įvairių triukšmų, dar nėra įmanoma išgauti tikslaus aplinkos žemėlapio ir tiksliai nustatyti roboto buvimo vietos. SLAM taikymą taip pat sunkina ir tai, kad aplinkos žemėlapio sudarymui ir svarbu žinoti tikslią roboto padėtį aplinkoje, o roboto padėties nustatymui yra svarbu žinoti tikslų aplinkos žemėlapį. Taip pat SLAM yra daugiau skirtas aplinkoje esančių objektų, tokių kaip sienos, kliūtys ir kt., aptikimui, nei paviršiaus savybių (paviršiaus nelygumų, pokrypių bei tvirtumo) atpažinimui (Tong *et al.* 2013; Olson, Agarwal 2013; Latif *et al.* 2013; Pfingsthorn, Birk 2013; Milford 2013).

1.3. Skaitiniai paviršiaus savybių vertinimo metodai

Dažniausiai naudojami paviršiaus nelygumo parametrai yra standartinė nuokrypa, standartinės nuokrypos pokytis, standartinės nuokrypos šlaitas, autokoreliacijos ilgis, efektyvusis šlaitas, medianinis ir absoliutusis šlaitai (Shepard *et al.* 2001).

Dažniausiai naudojamas ir lengviausiai apskaičiuojamas paviršiaus nelygumo parametras yra standartinė nuokrypa, angliškoje literatūroje dažniausiai vadinama RMSE (angl. *root-mean-square error*). Standartinės nuokrypos vertė ξ apskaičiuojama pagal (1.3) formulę.

$$\xi = \sqrt{\left[\frac{1}{n-1} \sum_{i=1}^{n} \left(z\left(x_i\right) - \overline{z} \right)^2 \right]}, \tag{1.3}$$

čia n yra taškų skaičius, $z\left(x_i\right)$ yra paviršiaus aukštis taške x_i, $\overline{z}$ yra visų taškų aukščio vidurkis.

Standartinės nuokrypos arba Allano pokytis (angl. *Allan variance*), *v*, yra išreiškiamas kaip standartinių nuokrypų aukščių skirtumas tarp dviejų taškų atskirtų per žingsnį, Δx, ir skaičiuojama pagal formulę (1.4).

$$v(\Delta x) = \sqrt{\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[z(x_i) - z(x_i + \Delta x) \right]^2 \right\}}. \tag{1.4}$$

Standartinės nuokrypos šlaitas s_{rms}, yra standartinės nuokrypos pokyčio, *v*, ir žingsnio, Δx, santykis (1.5).

$$s_{rms} = \frac{v(\Delta x)}{\Delta x}. \tag{1.5}$$

Dažniausiai standartinės nuokrypos šlaitas yra išreiškiamas kampu $\theta_{rms} = \tan^{-1}(s_{rms})$. Kaip ir standartinės nuokrypos pokytis, šlaitas (1.5) yra žingsnio funkcija. Šlaitas dažniausiai skaičiuojamas tik vienam žingsniui, dažniausiai mažiausiam gautam paviršiaus profilio bandinių žingsniui. Standartinės nuokrypos šlaitų pasiskirstymas apskaičiuotas išilgai paviršiaus profilio linijos yra įvardijamas vienakrypčiu. Priešingu atveju šlaitai atspindi maksimalų pokrypį (pvz. topografinio kontūro gradientą) kiekviename trimačio paviršiaus taške.

Paviršiaus profilio autokoreliacijos funkcija yra normalizuota kovariacija tarp profilio ir to paties, tik tam tikru žingsniu perstumto, profilio (Turcotte 1997). Autokoreliacija yra lygi 1, kai žingsnis yra lygus 0. Autokoreliacijos ilgis (dar vadinamas tiesiog koreliacijos ilgiu) *l*, yra dažnai apskaičiuojamas kaip atstumas arba žingsnis, reikalingas sumažinti normalizuotą koreliaciją iki 1/e arba –37 %. Lygesnių paviršių autokoreliacijos ilgis paprastai yra didelis, nelygių paviršių *l* yra labai mažas. Balto triukšmo autokoreliacijos ilgis yra lygus nuliui, o tiesios horizontalios linijos autokoreliacijos ilgis yra lygus begalybei.

Neretai paviršiaus nelygumas yra apibrėžiamas kaip standartinės nuokrypos ir autokoreliacijos ilgio proporcija. Dažnai pasitaiko, kad ši proporcija yra neteisingai įvardijama kaip standartinės nuokrypos šlaitas. Tam, kad būtų išvengta nesusipratimų, siūloma naudoti terminą „efektyvusis šlaitas“ (Campbell, Garvin 1993), skaičiuojamas pagal (1.6) formulę.

$$s_{eff} = \frac{\xi}{l}. \tag{1.6}$$

Kaip ir standartinės nuokrypos šlaitas, efektyvusis šlaitas dažnai išreiškiamas kampu: $\theta_{eff} = \tan^{-1}\left(s_{eff}\right)$.

Yra manoma, kad standartinės nuokrypos šlaitas prastai atspindi tikrąjį paviršiaus profilį, dėl to, kad nuošalūs taškai arba ištęstas šlaito dažnių pasiskirstymas paslinks standartinės nuokrypos šlaitą link didesnių verčių (Kreslavsky, Head 1999). Norint to išvengti, stačius šlaitus galima skaičiuoti kaip absoliutųjį šlaitą (1.7).

$$s_{abs} = \frac{1}{\Delta x}\left\{\frac{1}{n}\sum_{i=1}^{n}\left|z\left(x_i\right) - z\left(x_i + \Delta x\right)\right|\right\}. \tag{1.7}$$

Kita galimybė – skaičiuoti medianinį šlaitą s_{med}, kuris paprasčiausiai yra paviršiaus absoliučiųjų šlaitų vidurkis. Kaip ir su visais šlaitais, medianinis ir absoliutusis šlaitai dažniausiai išreiškiami kampais $\theta_{abs} = \tan^{-1}\left(s_{abs}\right)$, $\theta_{abs} = \tan^{-1}\left(s_{med}\right)$.

Neretai, norint įvertinti vien paviršiaus nelygumą, reikia keleto jį apibūdinančių parametrų. Dažniausiai pasitelkiami aukščiau išvardinti parametrai, tačiau jie gali būti pakeisti taip, kad geriausiai atitiktų keliamus reikalavimus. Castelnovi *et al.* 2005, eksperimentiškai nustatė, kad jų roboto valdymo sistemai užtenka tokių keturių pakeistų paviršiaus nelygumo parametrų:

RSB vidutinio nelygumo (angl. *Root Squared Bias Average Roughness*) – paviršiaus profilio amplitudės pokyčio pasiskirstymo, kaip aukščio funkcijos (1.8):

$$RSB = \sqrt{\frac{1}{120}\sum_{i=0}^{119}\left(height\left[i\right] - av_{lastline}\right)^2}. \tag{1.8}$$

AAS (angl. *Average Absolute Slope*) – absoliučiojo šlaito tarp dviejų nuskaitytų taškų vidurkio (1.9):

$$AAS = \frac{1}{120}\sum_{i=1}^{120}\left|height\left[i-1\right] - height\left[i\right]\right|. \tag{1.9}$$

RMSAS (angl. *Root-Mean-Square Average Slope*) – šlaito, naudoto AAS matavime, standartinės nuokrypos (1.10):

$$RMSAS = \sqrt{\frac{1}{120}\sum_{i=1}^{120}\left(height[i-1] - height[i]\right)^2}\ . \tag{1.10}$$

IRMS (angl. *Ideal Root-Mean-Square*) – ideliai lygaus paviršiaus standartinės nuokrypos, naudojamos kaip atskaitos taškas (1.11):

$$IRMS = \sqrt{\frac{1}{120}\sum_{i=0}^{119} height[i]^2}\ . \tag{1.11}$$

Kiekvienas standartinės nuokrypos parametras yra ribose tarp 0 ir maksimalios, eksperimentiškai nustatytos, vertės.

RSB vertė yra artima standartinės nuokrypos vertei, tik skiriasi tuo, kad RSB skaičiuojamas imant ne esamų taškų vidurkį, o ankstesnio matavimo. Ankstesnio vidurkio naudojimas leidžia įvertinti netolydumus judėjimo kryptimi, ko neleidžia kiti trys nelygumo parametrai.

Svarbu paminėti, kad visus šiuos keturis parametrus galima apskaičiuoti iš vieno lazerinio atstumo jutiklio matavimo rezultatų.

Paviršiaus praeinamumą galima apibūdinti trimis parametrais (Ishigami *et al.* 2011): paviršiaus nelygumo, pasvirimo ir kelio ilgio indeksais.

Paviršiaus nelygumo indeksas tiesiogiai nulemia pravažiuojamumą. Paviršiaus nelygumo indeksas B_{ij} yra paviršiaus iškilimų standartinė nuokrypa, roboto projekcijos regione (po robotu) R_{ij} (1.12), kuris priklauso nuo roboto matmenų.

$$B_{ij} = \sqrt{\frac{1}{n}\sum_{R_{ij}}\left(z\left(R_{ij}\right) - \overline{z}\left(R_{ij}\right)\right)^2}\ . \tag{1.12}$$

Paviršiaus pasvirimo indeksas atspindi ratų (kojų) praslydimą ant paviršiaus, taip pat riziką robotui apvirsti. Paviršiaus pasvyrimo indeksas yra išskaidomas į du kampus, šoninį (angl. *roll*) ir išilginį (angl. *pitch*). Jie apskaičiuojami kaip kampai tarp inercinės koordinatės ir pseudo plokštumos, sudarytos iš keturių roboto ratų kontaktų su paviršiumi projekcijos regione R_{ij}:

šoninis kampas: $\Theta_{xij} = \theta_x\left(R_{ij}\right)$;

išilginis kampas: $\Theta_{yij} = \theta_y\left(R_{ij}\right)$.

Indekso nustatymui parenkamas didžiausias kampas.

Kelio ilgio indeksas leidžia rasti trumpiasią kelią tarp esamos padėties ir tikslo. Kelio indeksas L_{ij} yra apskaičiuojamas kaip erdvinio vektoriaus ilgis tarp dviejų taškų (1.13).

$$L_{ij} = \left|\vec{n}_{ij}\right| = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2}\,. \tag{1.13}$$

Matomą paviršių suskaidžius į tam tikro dydžio segmentus, celes, kiekvienam segmentui galima įvertinti jo praeinamumą. Segmentų praeinamumas gali būti apibendrintas pavojaus koeficientu (angl. *danger value*) (Stelzer *et al.* 2012). Segmento pavojaus koeficientas, d ($d \in \{[0, 1], \infty\}$), apibūdina to segmento paviršiaus praeinamumo sudėtingumą. Segmentas laikomas praeinamu, jei paviršius jame nekelia kritinės rizikos roboto judėjimui. Pavojaus koeficientas $d = 0$ nurodo, kad paviršius segmente yra visiškai lygus ir robotas juo gali judėti visiškai laisvai. Segmentams priskiriamas didesnis pavojaus koeficientas, jei paviršius juose sunkiau praeinamas. Pavojaus koeficientas $d = 1$ nurodo, kad paviršius segmente yra labai sunkiai praeinamas. Nepraeinamo paviršiaus segmentams yra priskiriama $d = \infty$. Nežinomiems paviršiaus segmentams priskiriama $d = 1$.

Išskiriami trys potencialūs pavojai (Stelzer *et al.* 2012): statūs šlaitai (s), dideli paviršiaus nelygumai (r) ir aukšti slenksčiai (h). Jei bent vienas iš rizikos kriterijų viršija apibrėžtas kritines ribas, segmentas yra žymimas, kaip nepraeinamas. Kritinės vertės yra maksimalus šlaitas s_{crit}, didžiausias paviršiaus nelygumas r_{crit} ir aukščiausias slenkstis h_{crit}, kuriuos robotas gali įveikti neapsiversdamas ar užstrigdamas. Pavojaus koeficientas skaičiuojamas pagal (1.14) formulę.

$$d = \alpha_1 \frac{s}{s_{crit}} + \alpha_2 \frac{r}{r_{crit}} + \alpha_3 \frac{h}{h_{crit}}, \tag{1.14}$$

čia α_1, α_2 ir α_3 yra svorio koeficientai, kurių suma yra 1.

Segmento šlaitas s skaičiuojamas išvedus plokštumą segmento regione. Kampas tarp šios plokštumos statmens ir globalios koordinačių sistemos z ašies nurodo segmento šlaito pokrypį s. Paviršiaus nelygumas r apskaičiuojamas kaip paviršiaus aukščių, nuo išvestos plokštumos, standartinė nuokrypa.

Šlaito aukštis h skaičiuojamas dviem etapais. Pirmiausia, apskaičiuojami vietiniai aukščių skirtumai keliems segmentams tam tikro dydžio kvadrato formos regione visiems segmentams apskrtinimo formos regione. Jei maksimalus

aukščių skirtumas tarp bet kurio segmento tame regione ir centrinio regiono segmento yra didesnis nei h_{crit} ir *s* tarp dviejų atitinkamų taškų yra didesnis nei s_{crit}, tai maksimalus aukščių skirtumas yra išsaugomas kaip tarpinis centrinio segmento slenksčio aukštis. Tada, centrinio apskritimo formos regiono slenksčio aukštis apskaičiuojamas pagal (1.15) formulę:

$$h = \min\left(h_{\max}, h_{\max} \cdot \frac{n_{st}}{n_{crit}} \right), \tag{1.15}$$

čia $h_{\max}$ yra maksimalus tarpinis slenksčio aukštis apskritimo formos regione, n_{st} regionų skaičius apskritame regione, kurio tarpinis slenksčio aukštis yra didesnis nei h_{crit} ir n_{crit} yra galiojantis regionų skaičius (pvz. 50), kurių tarpinis slenksčio aukštis yra didesnis nei h_{crit}. Šis metodas taip pat leidžia nedidelius stačius šlaitus interpretuoti kaip laiptelius (Stelzer *et al.* 2012).

Aprašytas praeinamumo nustatymo metodas priklauso nuo daugelio parametrų: tokių kaip roboto dydžio, kritinio slenksčio aukščio, paviršiaus nelygumo ir šlaito verčių, bei trijų svorio koeficientų. Šie parametrai leidžia šį metodą pritaikyti skirtingiems robotams. Tačiau, nors kai kurie parametrai yra iš anksto žinomi ir priklauso nuo roboto konstrukcijos, kiti parametrai turi būti rasti empiriškai.

1.4. Pirmojo skyriaus išvados ir disertacijos uždavinių formulavimas

Atlikus literatūros analizę disertacijos tema pastebėta, kad šiuolaikiniuose robotuose aplinkos atpažinimui dažniausiai naudojamos dvi vaizdo kameros bei lazeriniai atstumo jutikliai, kurių informacijos apdorojimui reikia sparčių valdiklių ar kompiuterių. Taip pat nėra visiškai aiškūs paviršiaus savybių fiziniai dydžiai, pagal kuriuos paviršius yra klasifikuojamas. Todėl disertacijos įvade iškelti uždaviniai lieka neišspręsti ir aktualūs.

1. Išanalizavus literatūrą galima teigti, kad paviršiaus savybių atpažinimas ir vertinimas yra viena iš aktualiausių robotų inžinerijos problemų, kurios sprendimas leis mobilius robotus taikyti įvairesnėse srityse.
2. Dažniausiai paviršiaus savybių atpažinimui naudojamos dvi vaizdo kameros bei lazeriniai atstumo jutikliai. Tačiau iš jų gautos informa-

cijos norint gauti reikiamą paviršiaus įvertinimą, tą informaciją reikia apdoroti sudėtingais algoritmais, dėl ko turi būti naudojami spartūs kompiuteriai.

3. Iš antros išvados seka, kad robotai, aplinkos atpažinimui, naudojantys minėtas priemones, negali veikti realiu laiku ir autonomiškai. Norint, kad robotas veiktų realiu laiku, informacija, gaunama iš kamerų arba lazerinių atstumo jutiklių, turi būti apdorojama išoriniame kompiuteryje ir rezultatai siunčiami atgal į roboto valdymo sistemą. Norint, kad robotas veiktų autonomiškai, robotas netenka galimybės realiu laiku atpažinti aplinkos ir priimti reikiamų sprendimų.
4. Dvi vaizdo kameros yra universaliausia paviršiaus savybių atpažinimo priemonė, kadangi leidžia atpažinti visas reikiamas paviršiaus savybes, tai yra paviršiaus nelygumus, paviršiaus šlaitus bei paviršiaus tvirtumą. Tuo tarpu, lazeriniai atstumo jutikliai labiausiai tinkami tik paviršiaus nelygumų ir šlaitų atpažinimui, tačiau iškart pateikia informaciją apie atstumus iki paviršiaus.
5. Vaizdų apdorojimu bei lazeriniu skenavimu pagrįsti aplinkos atpažinimo metodai geri tolimesnių veiksmų planavimui, tačiau operatyviam daugiakojo roboto judesių valdymui priimtinesni algoritmai, įvertinantys paviršiaus savybes esamoje vietoje pagal taktilinių ir inercinių jutiklių duomenis.
6. Paprasčiausias paviršiaus nelygumų vertinimo metodas yra standartinė nuokrypa. Šis metodas leidžia vienu skaičiumi nusakyti paviršiaus nelygumų dydį ir turi paprastą matematinę išraišką. Didžiausias šio metodo trūkumas yra tai, kad pakankamam tikslumui gauti yra reikalingas didelis matavimų skaičius.

Darbo tikslui pasiekti darbe reikia spręsti šiuos uždavinius:

1. Sudaryti roboto imitacinį ir fizinį modelius bei sudaryti trajektorijų generavimo matematines išraiškas adaptyviam roboto ėjimui.
2. Ištirti roboto ėjimą lygiu paviršiumi, taikant adaptyvią eiseną ir įvertinti paviršiaus atpažinimo nuokrypas.
3. Sudaryti ir ištirti algoritmą, leidžiantį atpažinti paviršiaus nelygumo savybes pagal esamas roboto pėdų koordinates.
4. Sudaryti ir ištirti šešiakojo žingsniuojančio roboto eisenos parinkimo pagal paviršiaus savybes algoritmą.

2

Šešiakojo roboto modelio sudarymo metodika

Skyriuje pateikiamas trumpas šešiakojo žingsniuojančio roboto modelio aprašymas. Sudaromas kinematinis modelis ir roboto pėdų trajektorijų matematinės išraiškos, kurios leidžia paprastai keisti roboto eiseną ir prisitaikyti prie paviršiaus nelygumų. Pagal sudarytą kinematinį modelį sudaromas roboto imitacinis modelis MATLAB® aplinkoje, kuris leidžia greitai patikrinti ir išbandyti roboto judėjimą bei skirtingus valdymo algoritmus.

Aprašomas roboto valdymo sistemos sudarymas ir algoritmas, kuris leidžia robotą valdyti paprastomis aukšto lygio komandomis, nesutrikdant roboto ėjimo.

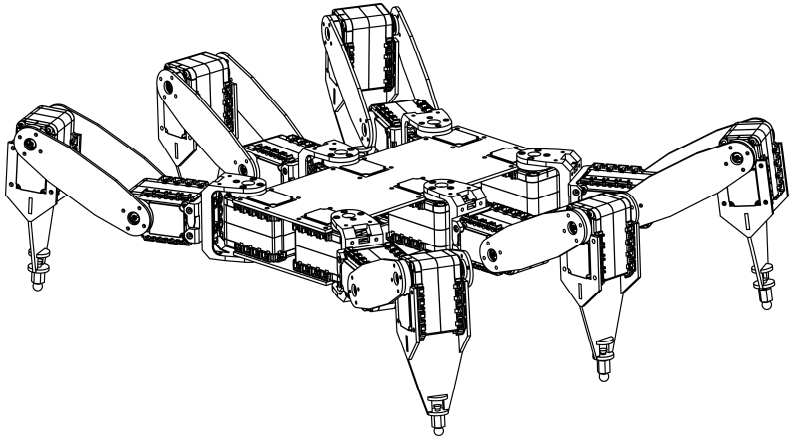
Taip pat pateikiami paviršiaus savybių vertinimo algoritmų aprašymai, kurie paremti roboto pėdų aukščio koordinačių vidutinės kvadratinės nuokrypos σ verte bei dviejų plokštumų tarp roboto pėdų pokrypio kampais.

Skyriaus tematika paskelbti keturi autoriaus straipsniai (Luneckas 2010; Luneckas, Udris 2011; Luneckas, Udris 2012; Luneckas, Udris 2013).

2.1. Kinematinio ir imitacinio roboto modelio sudarymas

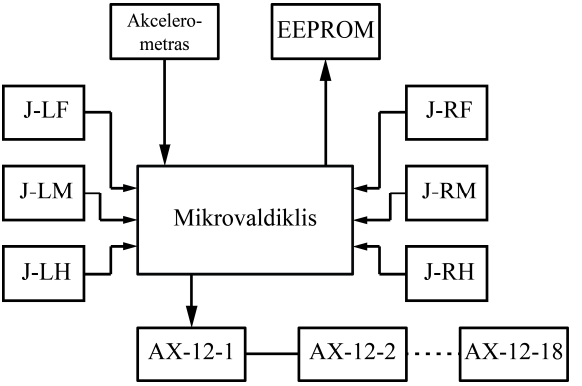
2.1.1. Roboto aprašymas

Roboto maketo judesių valdymui panaudoti Dynamixel® firmos AX-12 servomechanizmai. Roboto dalys suprojektuotos programiniu paketu *AutoCAD®*, roboto konstrukcinis brėžinys sudarytas programiniu paketu *Solidworks®* (2.1 pav.).

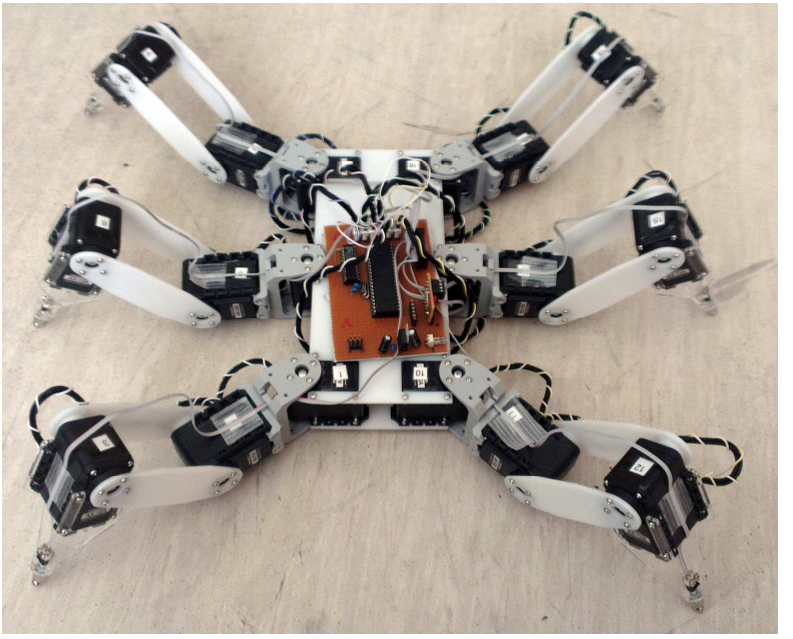


2.1 pav. Šešiakojo roboto *Solidworks®* brėžinys
Fig. 2.1. Hexapod robot draft in *Solidworks®*

Servomechanizmai valdomi nuosekliąja asinchronine sąsaja (USART) specializuotu protokolu, todėl labai supaprastėja roboto valdymo sistema, nes užtenka tik vienos duomenų linijos visoms pavaroms. Taip pat labai sumažėja krūvis mikrovaldikliui, kuriam nereikia generuoti pavarų posūkio kampo valdymo signalų, o pakanka išsiųsti atitinkamas komandas pavaroms. Prie valdiklio prijungti kiekvienos pėdos taktiliniai jutikliai, indikuojantys ar koja yra pastatyta ant paviršiaus, taip pat per I2C sąsają prijungti akcelerometras bei pastovioji duomenų atmintinė (EEPROM). Valdymui naudojamas Atmel® firmos mikrovaldiklis Atmega16 su 16 MHz taktų dažnio generatoriumi. Roboto valdymo sistemos funkcinė schema pateikta 2.2 paveiksle. Fizinis šešiakojo roboto maketas pavaizduotas 2.3 paveiksle.



2.2 pav. Roboto valdymo sistemos funkcinė schema
Fig. 2.2. Function diagram of robot‘s control system

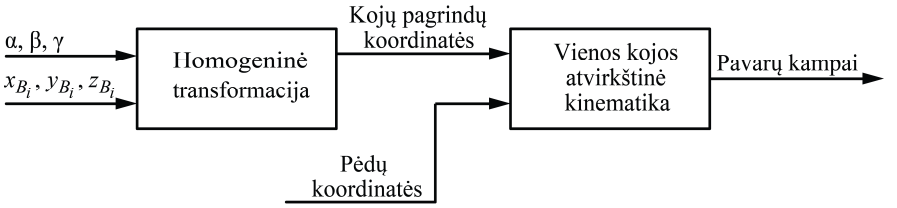


2.3 pav. Šešiakojo roboto maketas
Fig. 2.3. Hexapod robot model

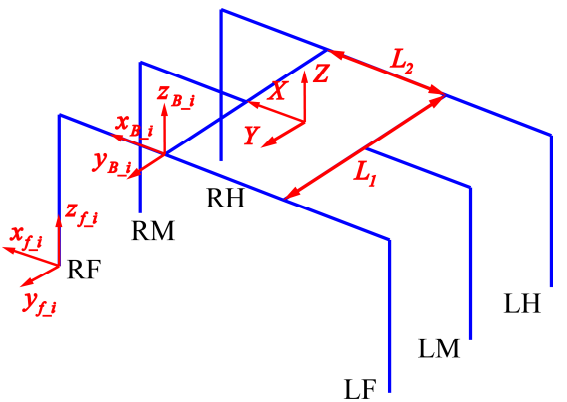
2.1.2. Kinematinio roboto modelio sudarymas

Norint palengvinti roboto kinematikos aprašymą, tikslinga aprašyti atskirų roboto dalių kinematiką ir ją vėliau sujungti į vieną visumą. Šešiakojo roboto kinematiką patogu nagrinėti atskyrus roboto kūno ir kojų kinematiką. Roboto kinematinio modelio struktūra pavaizduota 2.4 paveiksle. Sudarant roboto kinematinį modelį yra svarbu teisingai prisikirti koordinačių sistemas, kurios leis sujungti visą roboto kinematiką į vieną bendrą sistemą.

Pagrindinė koordinačių sistema yra roboto kūno koordinačių sistema, visos kitos koordinatės yra transformuojamos į ją. Kitos reikalingos koordinačių sistemos yra priskiriamos kiekvienos kojos pagrindui bei pėdai (2.5 pav.).



2.4 pav. Roboto kinematinio modelio struktūra
Fig. 2.4. Structure of robot's kinematic model



2.5 pav. Roboto kinematinė schema su priskirtomis koordinačių sistemomis
Fig. 2.5. Robot's kinematic diagram with assigned coordinate frames

2.5 paveiksle pavaizduotos tokios robotui priskirtos koordinačių sistemos: XYZ – kūno koordinačių sistema (pagrindinė), $x_{B_i} y_{B_i} z_{B_i}$ – kojos pagrindo ko-

ordinačių sistema, $x_{f_i} y_{f_i} z_{f_i}$ – pėdos koordinačių sistema, i – kojos indeksas, L_1 – roboto kūno ilgis, L_2 – roboto kūno plotis.

Roboto kūno padėtį galima aprašyti šešiais parametrais: trys posūkio kampai (α, β, γ) apie roboto kūno koordinačių ašis X, Y, Z ir trys poslinkiai šiomis ašimis.

Pirmiausia reikia transformuoti kojų pagrindų koordinačių sistemas į roboto kūno koordinačių sistemą, tai leis žinoti kiekvienos kojos pagrindo koordinates roboto koordinačių sistemoje. Tai gali būti atlikta naudojant homogenines transformacijų matricas. Vienos kojos pagrindo transformavimui yra reikalinga viena posūkio matrica, aprašanti posūkius apie visus minėtus kampus, ir viena postūmio matrica. Bendru atveju homogeninė transformacijos matrica pateikta (2.1) formulėje (Jazar 2010; Bajd *et al.* 2010).

$$\mathbf{B}_i = \mathbf{T}_{XYZ} \cdot \mathbf{T}_i, \tag{2.1}$$

čia:

$$\mathbf{T}_{XYZ} \equiv \left[\begin{array}{ccc|c} & \mathbf{R}_{XYZ} & & \mathbf{d}_{XYZ} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & X_0 \\ r_{21} & r_{22} & r_{23} & Y_0 \\ r_{31} & r_{32} & r_{33} & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.2}$$

$$\mathbf{B}_i = \begin{bmatrix} X_{B_i} \\ Y_{B_i} \\ Z_{B_i} \\ 1 \end{bmatrix}, \tag{2.3}$$

$$\mathbf{T}_i = \begin{bmatrix} x_{B_i} \\ y_{B_i} \\ z_{B_i} \\ 1 \end{bmatrix}, \tag{2.4}$$

$$\mathbf{d}_{XYZ} = \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix}. \tag{2.5}$$

$X_{B_i}, Y_{B_i}, Z_{B_i}$ (2.3) yra kojų pagrindų koordinatės roboto kūno koordinačių sistemoje atlikus transformaciją, $x_{B_i}, y_{B_i}, z_{B_i}$ (2.4) – kojų pagrindų koordinačių sistemų poslinkiai *x*, *y* ir *z* ašimis, X_0, Y_0, Z_0 (2.5) – roboto kūno koordinačių sistemos centro koordinatės, $\mathbf{R}_{XYZ}$ – posūkio apie visas tris ašis matrica.

Posūkio matrica $\mathbf{R}_{XYZ}$ (2.6) gaunama sudauginus tris posūkio matricas (2.7) –(2.9) apie kiekvieną roboto kūno koordinačių sistemos ašį (Jazar 2010; Bajd *et al.* 2010; Baškys, Fedaravičius 2004; Zielinska 2005).

$$\mathbf{R}_{XYZ} = \mathbf{R}_X \cdot \mathbf{R}_Y \cdot \mathbf{R}_Z, \tag{2.6}$$

$$\mathbf{R}_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}, \tag{2.7}$$

$$\mathbf{R}_Y = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}, \tag{2.8}$$

$$\mathbf{R}_Z = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.9}$$

Poslinkio matricos $\mathbf{T}_i$ (2.4) sudaromos kiekvienos kojos pagrindui atskirai (2.10)–(2.15).

$$\mathbf{T}_{\mathrm{RF}} = \begin{bmatrix} L_2/2 \\ L_1/2 \\ 0 \\ 1 \end{bmatrix}, \tag{2.10}$$

$$\mathbf{T}_{\mathrm{RM}} = \begin{bmatrix} L_2 / 2 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2.11}$$

$$\mathbf{T}_{\mathrm{RH}} = \begin{bmatrix} L_2 / 2 \\ -L_1 / 2 \\ 0 \\ 1 \end{bmatrix}, \tag{2.12}$$

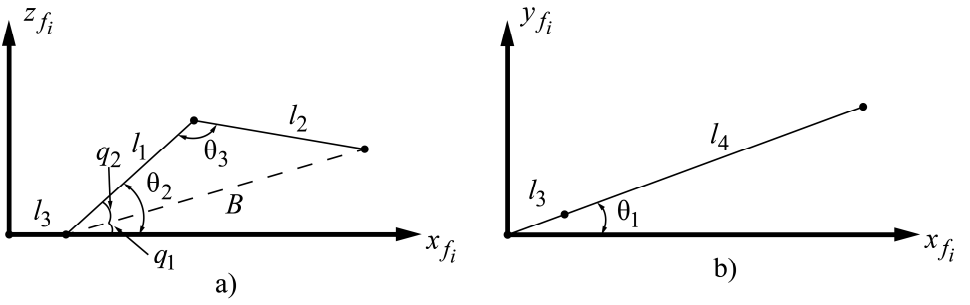
$$\mathbf{T}_{\mathrm{LF}} = \begin{bmatrix} -L_2 / 2 \\ L_1 / 2 \\ 0 \\ 1 \end{bmatrix}, \tag{2.13}$$

$$\mathbf{T}_{\mathrm{LM}} = \begin{bmatrix} -L_2 / 2 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2.14}$$

$$\mathbf{T}_{\mathrm{LH}} = \begin{bmatrix} -L_2 / 2 \\ -L_1 / 2 \\ 0 \\ 1 \end{bmatrix}. \tag{2.15}$$

Norint nustayti reikiamą pėdos padėtį erdvėje, reikia apskaičiuoti jos pavarų posūkio kampus. Šie posūkio kampai randami išsprendus atvirkštinės kinematikos uždavinį vienai kojai (Hooper 2009).

Sprendžiant atvirkštinės kinematikos uždavinį geometriniu metodu (Botello *et al.* 2010; Petrov, Dimitrov 2010; Manoiu-Olaru, Nitulescu 2011; Netto *et al.* 2000), reikalingos kojos projekcijos *xy* (2.6 pav., a) ir *xz* (2.6 pav., b) plokštumose.



2.6 pav. Kojos projekcijos *xy* (a) ir *xz* (b) plokštumose
Fig. 2.6. Leg projections into *xy* and *xz* planes

$$\theta_1 = \arctan\left(\frac{y}{x + l_3}\right), \tag{2.16}$$

$$\theta_2 = \underbrace{\arccos\left(\frac{l_1^2 - l_2^2 + B^2}{2 \cdot l_1 \cdot B}\right)}_{q_1} + \underbrace{\arcsin\left(\frac{z}{B}\right)}_{q_2}, \tag{2.17}$$

$$\theta_3 = \arccos\left(\frac{l_1^2 + l_2^2 - B^2}{2 \cdot l_1 \cdot l_2}\right), \tag{2.18}$$

$$B = \sqrt{l_4^2 + z^2}\,, \tag{2.19}$$

$$l_4 = \sqrt{x^2 + y^2}\,, \tag{2.20}$$

čia θ_1 – klubo (angl. *coxa*) pavaros kampas (2.16), θ_2 – šlaunies (angl. *femur*) pavaros kampas (2.17), θ_3 – alkūnės (angl. *tibia*) pavaros kampas (2.18), B – pagalbinė menamoji linija, jungianti šlaunies pradžią ir blauzdos pabaigą (2.19), l_1 – šlaunies ilgis, l_2 – blauzdos ilgis, l_3 – atstumas tarp klubo ir šlaunies ašių, l_4 – linijos B projekcijos į *xy* plokštumą ilgis (2.20).

Norint, kad kojos pagrindas judėtų viena kryptimi, atitinkama pėda turi judėti priešinga kryptimi. Kai robotas stovi neutralioje būsenoje, esamos pėdų koordinatės jų kojos pagrindo koordinačių sistemoje randamos pagal (2.21)–(2.23).

$$x_i = l_2 + x_{f_i} - (x_{B_i} + x_{B_i\_offset}), \tag{2.21}$$

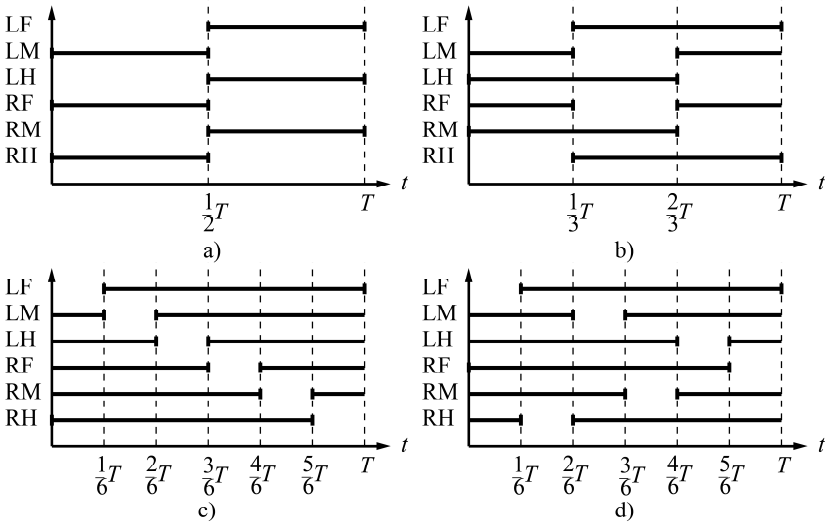
$$y_i = y_{f_i} - (y_{B_i} + y_{B_i\_offset}), \tag{2.22}$$

$$z_i = -l_3 + z_{f_i} - (z_{B_i} + z_{B_i\_offset}), \tag{2.23}$$

čia x_i, y_i, z_i – tikrosios pėdų koordinatės, naudojamos atvirkštinės kinematikos skaičiavimuose, $x_{f_i}, y_{f_i}, z_{f_i}$ – norimos pėdos koordinatės, $x_{B_i}, y_{B_i}, z_{B_i}$ – norimos kojos pagrindo koordinatės, $x_{B_i\_offset}, y_{B_i\_offset}, z_{B_i\_offset}$ – kojos pagrindo koordinačių sistemos poslinkiai.

2.1.3. Pėdų trajektorijų generavimo matematinių išraiškų sudarymas

Norint užtikrinti sklandų šešiakojo roboto judėjimą, reikia sinchronizuoti visų jo kojų judesius. Kaip turi būti sinchronizuotos roboto kojos einant skirtingomis eisenomis parodo laikinės eisenų diagramos (Siciliano, Khatib 2008; Kar *et al.* 2003; Ridderström 2003) (2.7 pav.).



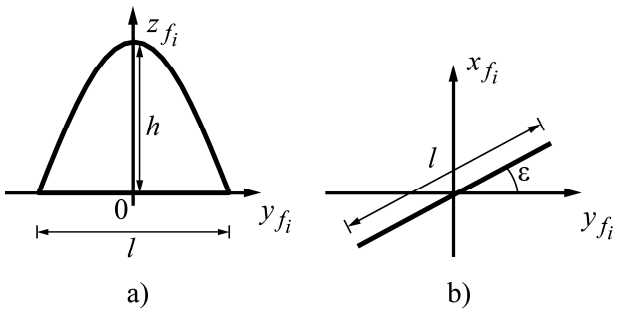
2.7 pav. Šešiakojo roboto eisenų diagramos: a) trikojė eisena, b) keturkojė eisena, c) banguojanti eisena, d) pulsuojanti eisena
Fig. 2.7. Hexapod robot's gait diagrams: a) tripod gait, b) tetrapod gait, c) wave gait, d) ripple gait

Visas roboto žingsnis yra kai jis perkelia visas šešias kojas į naują padėtį. Priklausomai nuo eisenos, vienas žingsnis trunka tam tikrą laiką. Laikas per kurį robotas įvykdo vieną žingsnį vadinamas žingsnio periodu (*T*). Laikas, kai koja turi būti perkeliama į naują padėtį, vadinamas tos kojos faze (φ_i). Kiekvienos eisenos periodai ir kojų fazės pateikiami 2.1 lentelėje.

2.1 lentelė. Eisenų periodai ir kojų perkėlimo fazės
Table 2.1. Gait periods and leg swing phases

Eisena	T	φ_{LF}	φ_{LM}	φ_{LH}	φ_{RF}	φ_{RM}	φ_{RH}
Banguojanti	6	0	1	2	3	4	5
Pulsuojanti	6	0	2	4	5	3	1
Keturkojė	3	0	1	2	1	2	0
Trikojė	2	0	1	0	1	0	1

Norint laisvai keisti roboto judėjimą, turi būti sudaryta galimybė keisti roboto žingsnio ilgį bei aukštį, taip pat kryptį, kuri nulems roboto judėjimo kryptį. Pėdų trajektorijos projekcijos pavaizduotos 2.8 paveiksle.



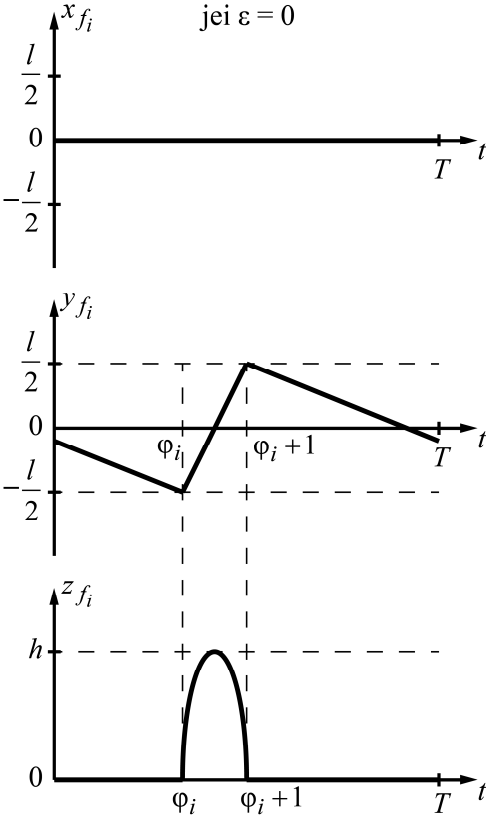
2.8 pav. Roboto pėdų trajektorijos projekcijos į *yz* (a) ir *xy* (b) plokštumas.
l – žingsnio ilgis, *h* – žingsnio aukštis, ε – roboto ėjimo kryptis
Fig. 2.8. Robot's leg trajectory projections into *yz* (a) and *xz* (b) planes.
l – step length, *h* – step height, ε – stride direction

Kaip matyti iš 2.8 paveikslo trajektorija yra atskirų pėdos koordinačių sinchroniškas kitimas tam tikrais dėsniais. Visi judesiai yra periodiški, todėl norint išgauti tokį roboto pėdų judesį, reikia išreikšti kiekvienos koordinatės kitimo laike dėsnius. Kiekvienos koordinatės laikinės funkcijos bendru atveju pateiktos 2.9 paveiksle.

Kiekvienos koordinatės funkcijas galima išskaidyti į tris periodus:

- $t \le \varphi_i$;
- $\varphi_i < t \le \varphi_i + 1$;
- $t > \varphi_i + 1$.

x ir *y* koordinačių funkcijų išraiškos yra identiškos ir gaunamos aprašant jas tiesės lygtimis per du taškus pateiktuose trijuose perioduose. *z* koordinatė antrame periode kinta sinuso dėsniu, o kituose perioduose yra lygi 0.



2.9 pav. Roboto pėdų koordinačių laikinės funkcijos
Fig. 2.9. Robot's leg coordinate time functions

Roboto pėdų koordinates galima generuoti taikant (2.24)–(2.26) išraiškas.

$$x_{f_i}(t)=\begin{cases}-\dfrac{l\cdot\sin(\varepsilon)\cdot(T-2\cdot\varphi_i+2\cdot t-1)}{2\cdot(T-1)}, & \text{kai } t\le\varphi_i,\\ (l\cdot\varphi_i-0{,}5)\cdot\sin(\varepsilon), & \text{kai } \varphi_i<t\le\varphi_i+1,\\ \dfrac{l\cdot\sin(\varepsilon)\cdot(T+2\cdot\varphi_i-2\cdot t+1)}{2\cdot(T-1)}, & \text{kai } t>\varphi_i+1.\end{cases} \quad (2.24)$$

$$y_{f_i}(t)=\begin{cases}-\dfrac{l\cdot\cos(\varepsilon)\cdot(T-2\cdot\varphi_i+2\cdot t-1)}{2\cdot(T-1)}, & \text{kai t}\le\varphi_i,\\ (l\cdot\varphi_i-0{,}5)\cdot\cos(\varepsilon), & \text{kai } \varphi_i<t\le\varphi_i+1,\\ \dfrac{l\cdot\cos(\varepsilon)\cdot(T+2\cdot\varphi_i-2\cdot t+1)}{2\cdot(T-1)}, & \text{kai } t>\varphi_i+1.\end{cases} \quad (2.25)$$

$$z_{f_i}(t)=\begin{cases}0, & \text{kai t}\le\varphi_i \text{ ir } t>\varphi_i+1,\\ h\cdot\sin\left((t-\varphi_i)\cdot\pi\right), & \text{kai } \varphi_i<t\le\varphi_i+1.\end{cases} \quad (2.26)$$

Įstačius T ir φ_i vertes gaunamos konkrečios kiekvienos pėdos koordinačių trajektorijų išraiškos skirtingoms eisenoms (Hidekazu, Hitoshi 2010; Estremera, de Santos 2005; Erden, Leblebicioğlu 2007; Kale *et al.* 2013; Hirose *et al.* 2013).

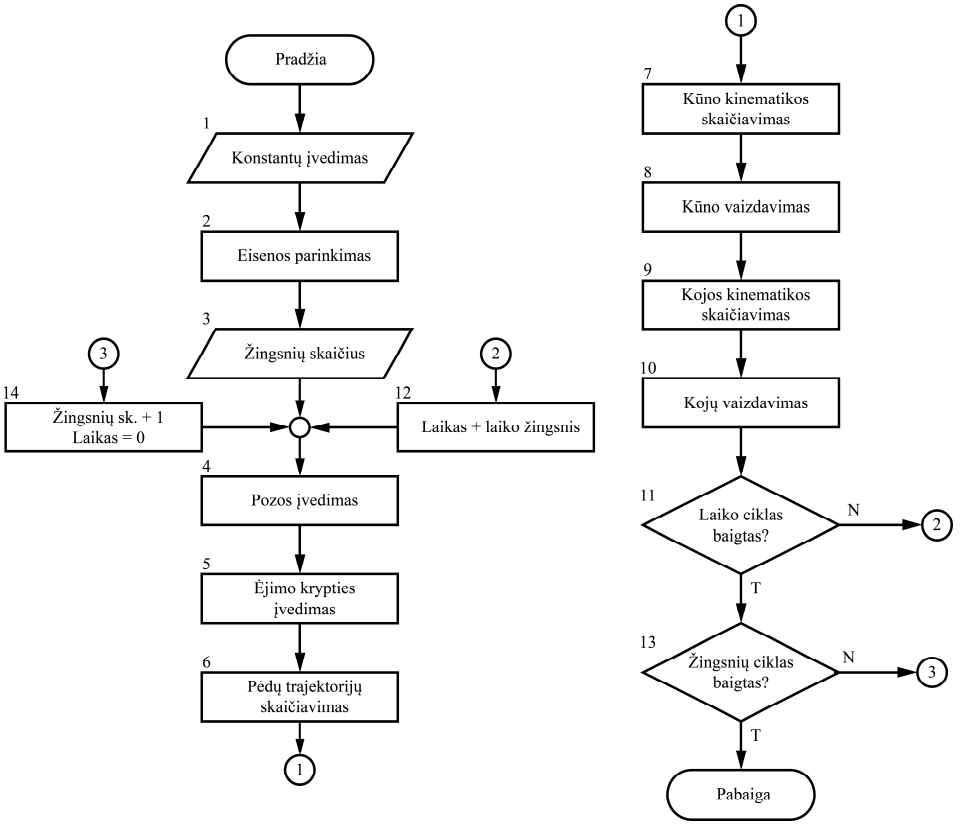
2.1.4. Imitacinio roboto modelio sudarymas

Šešiakojo roboto judesių sudėtingumas (Juang *et al.* 2010; Manoiu-Olaru, Nitulescu 2011) reikalauja, kad visa sistema būtų pirmiau patikrinta imituojant ją virtualioje aplinkoje. Šiam tikslui pasirenkamas MATLAB® programinis paketas. Roboto imitacinis modelis sudaromas remiantis kinematiniu modeliu. Pirmas modeliavimo tikslas yra patikrinti sudaryto kinematinio modelio teisingumą. Roboto imitacinis modelis sudaromas atsižvelgiant į jo vėlesnį pritaikymą kuriant roboto valdymo programą. Todėl roboto modelis sudaromas MATLAB® tekstine programavimo kalba. Roboto modelio programa turi būti kuo artimesnė būsimai roboto valdymo programai, kuri rašoma C kalba. Paprastesnei modelio analizei bei patikrai sudaroma galimybė roboto judesius vizualizuoti.

Roboto imitacinio modelio, o kartu ir valdymo programos tikslas visus kinematinius skaičiavimus suvesti į roboto pavarų kampų vertes. Todėl patogu visas pavarų vertes laikyti viename masyve, iš kurio būtų labai paprasta bet kuriuo momentu išrinkti reikiamą vertę.

Pirmiausia reikia apibrėžti konstantas, kurios yra naudojamos tolimesniuose skaičiavimuose. Šios konstantos yra roboto matmenys (roboto kūno, kojų), taip pat žingsnio ilgis ir aukštis (2.10 pav., 1). Visi matmenys nurodomi milimetrais.

Eisenos parinkimui naudojamas *string* tipo kinamasis `r_gait`. Jis gali įgyti vieną iš keturių verčių: `wave`, `ripple`, `bipod`, `tripod`. Įvesta eisenos kintamojo vertė yra palyginama su kiekviena iš simbolių eilučių esančių sąlygoje `if`. Palyginimui naudojama komanda `strcmp()`. Sutapus vertėms programa atitinkamiems kintamiesiems priskiria eisenos periodą ir kojų fazes (2.10 pav. 2).



2.10 pav. Imitacinio roboto modelio algoritmas
Fig. 2.10. Algorithm of robot‘s imitation program

Žingsnių skaičius `step_no` nurodo kiek pilnų žingsnių turi atlikti robotas (2.10 pav., 3). Minimali vertė yra 1. Įvedus visu reikiamus pradinius duomenis

programa pereina į du `for` ciklus, kurie yra kartojami kol bus įvykdyti visi roboto žingsniai, nurodyti `step_no` kintamajame. Pirmasis `for` ciklas yra skirtas žingsnių skaičiaus patikrinimui ir didinimui, antrasis yra elementarių laiko tarpsnių ciklas, kuris mažais intervalais atlieka roboto kinematikos skaičiavimus per visą vieno pilno žingsnio periodą. Antrojo `for` ciklo pradžioje ir prieš visus kinematinius skaičiavimus, turi būti įvedami šeši roboto kūno parametrai, trys posūkio kampai (`a, b, c`) ir trys poslinkiai (`y_B x_B z_B`) (2.10 pav., 4). Posūkio kampai įvedami laipsniais, poslinkiai – milimetrais. Norint keisti roboto ėjimo kryptį, reikia įvesti ėjimo krypties kampą `dir` (2.10 pav., 5), kuris naudojamas pėdų trajektorijų skaičiavimuose. Kampas įvedamas laipsniais. Parinkus eiseną ir kitus parametrus turi būti suskaičiuojamos reikiamų pėdų koordinatės (2.10 pav., 6). Koordinatės skaičiuojamos kiekvienai pėdai atskirai, kreipiantis į trajektorijų generavimo funkciją šešis kartus, perduodant jai atitinkamus parametrus ir funkcija kaskart grąžina konkrečios pėdos koordinates.

```
[RF_x, RF_y, RF_z] = feet_traj(RF_ph, T, l, h, t, dir);
```

Aukščiau pateiktas kreipimasis į funkciją, kuri atgal į imitacinę programą grąžina dešinės priekinės pėdos koordinates, esamu laiko momentu. Galima pastebėti, kad į trajektorijų generavimo funkciją yra perduodami tokie parametrai:

- `RF_ph` – pėdos fazė;
- `T` – eisenos periodas;
- `l` – žingsnio ilgis;
- `h` – žingsnio aukštis;
- `t` – einamas laikas;
- `dir` – roboto ėjimo krypties kampas.

Visos reikiamos roboto pėdų koordinatės yra suvedamos į šešis skirtingus masyvus. Kiekvienas masyvas atitinka kiekvieną koją, o jo kintamieji yra pėdos koordinatės. Tai supaprastina ir palengvina pėdų koordinačių keitimą.

```
%reikiamos pėdų koordinatės
%foot_coord =[x y z]
RF_coord = [RF_x RF_y RF_z];
RM_coord = [RM_x RM_y RM_z];
RH_coord = [RH_x RH_y RH_z];
LF_coord = [LF_x LF_y LF_z];
LM_coord = [LM_x LM_y LM_z];
LH_coord = [LH_x LH_y LH_z];
```

Nors pėdų koordinačių įvedimas yra paprastesnis į skirtingus masyvus, tačiau nuskaitant jas yra itin nepatogu kaskart kreiptis į skirtingą masyvą. Todėl atskiri visų pėdų koordinačių masyvai yra suvedami į vieną bendrą masyvą.

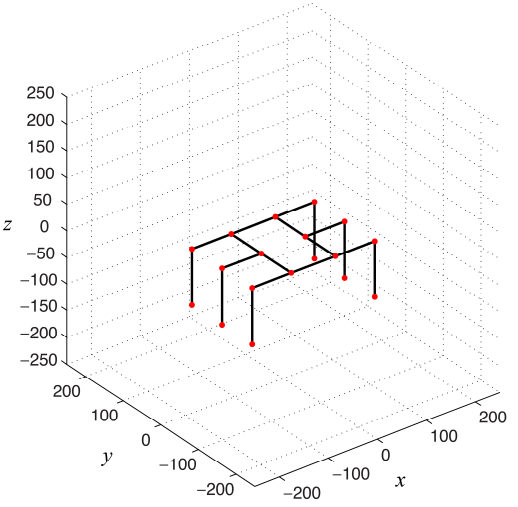
```
feet_coord = [RF_coord; RM_coord; RH_coord;
              LF_coord; LM_coord; LH_coord];
```

Tokiu būdu, norint nuskaityti kojų pėdų koordinates tereikia kreiptis į vieno masyvo elementus. Pavyzdžiui norint nuskaityti kairės vidurinės (LM) kojos *x* koordinatę, į masyvą reikia kreiptis taip: `feet_coord(5, 1);`. Sekantis žingsnis yra roboto kūno kinematikos skaičiavimas pagal (2.6)–(2.15) formules (2.10 pav., 7). Gaunamos kojų pagrindų koordinatės taip pat kaip ir pėdų koordinatės suvedamos į vieną bendrą masyvą:

```
%kojų pagrindų koordinatės sudedamos į bendrą masyvą
%stulpeliai yra koordinatės - x y z
%eilutės yra kojos - RF RM RH LF LM LH
leg_base_coord = [B_RF(1,4) B_RF(2,4) B_RF(3,4);
                  B_RM(1,4) B_RM(2,4) B_RM(3,4);
                  B_RH(1,4) B_RH(2,4) B_RH(3,4);
                  B_LF(1,4) B_LF(2,4) B_LF(3,4);
                  B_LM(1,4) B_LM(2,4) B_LM(3,4);
                  B_LH(1,4) B_LH(2,4) B_LH(3,4);];
```

Šis masyvas yra 6x3 matrica, kurios eilutės yra kojos, o stulpeliai – koordinatės.

Kūno vaizdavimas atliekamas naudojant `line([x1 x2], [y1 y2], [z1 z2]);` funkciją (2.10 pav., 8). Roboto kūną sudaro šešios linijos, kurios jungia visus kojų pagrindus (2.11 pav.). Kojų pagrindų koordinatės paimamos iš `leg_base_coord` masyvo.

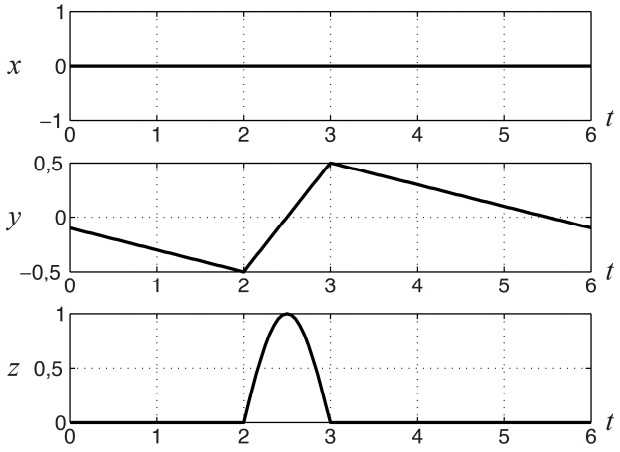


2.11 pav. Roboto imitacinio modelio vaizdas. Taškai žymi vietas per kurias piešiamas visas roboto vaizdas

Fig. 2.11. Vizualisation of robot‘s imitation. Dots represent line points for drawing robot

Visų kojų pavarų kampai skaičiuojami remiantis atvirkštinės kinematikos vienai kojai išraiškomis (2.16)–(2.20) (2.10 pav., 9). Gauti pavarų kampai įrašomi į bendrą visų pavarų masyvą `motor_angles()`. Šis masyvas yra 6x3 matrica, kurios eilutės atitinka kojas, o stulpeliai – pavaras. Kampai šiame masyve išsaugomi laipsniais. Kojų animacija (2.10 pav., 10), analogiškai kaip ir kūno, atliekama naudojant `line` funkciją (2.11 pav.). Elementarių laiko tarpsnių ciklo pabaigoje (2.10 pav., 11) atliekama patikra ar esamas laikas yra lygus esamos eisenos periodui, jei ne, tada laiko kintamasis yra didinamas laiko žingsniu (2.10 pav., 12) ir visi skaičiavimai yra atliekamai iš naujo, jei esamas laikas yra lygus eisenos periodui, programa tikrina ar atlikti visi žingsniai (2.10 pav., 13), jei ne, žingsnių kintamasis didinamas vienetu (2.10 pav., 14) ir laiko ciklas kartu su skaičiavimaid paleidžiamas iš naujo. Jei atlikti visi žingsniai, imitacinė programa stabdoma.

Trajektorijų generavimo išraiškos patikrinamos suskaičiuojant eisenų taškus visam eisenos periodui. 2.12 paveiksle pateiktos dešinės vidurinės pėdos sugeneruotos koordinatės banguojančiai eisenai, kai žingsnio aukštis ir ilgis yra lygūs 1, o ėjimo kryptis (kampas ε) lygi 0.

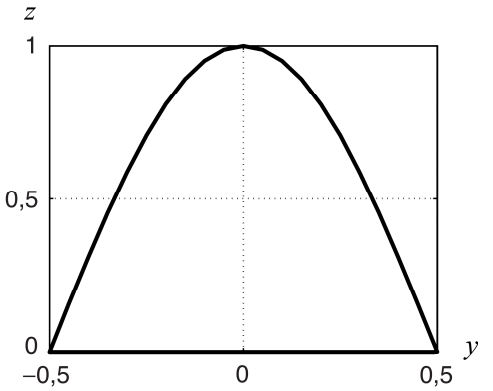


2.12 pav. Sugeneruotos dešinės vidurinės pėdos koordinatės banguojančiai eisenai

Fig. 2.12. Generated right middle foot trajectory coordinates for wave gait

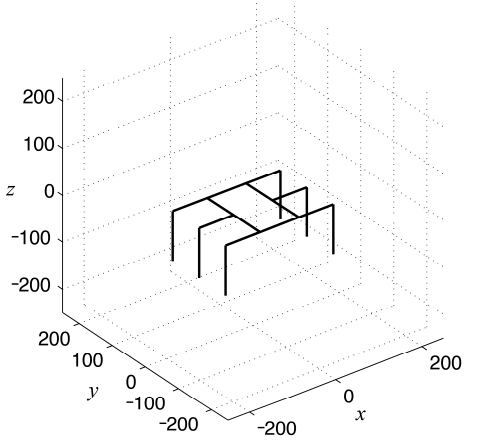
Kaip matyti iš 2.12 ir 2.9 paveikslėlių, sugeneruotos koordinatės visiškai sutampa su norimomis koordinačių išraiškomis. Taip pat, žiūrint į trajektoriją *yz* projekcijoje (2.13 pav.), matyti, kad jos vaizdas taip pat visiškai sutampa su norimos trajektorijos forma (2.8 pav.). Taigi, galima tvirtai teigti, kad trajektorijų

generavimo matematinės išraiškos yra teisingos ir užtikrins sinchroniškus ir teisingus roboto kojų judesius.



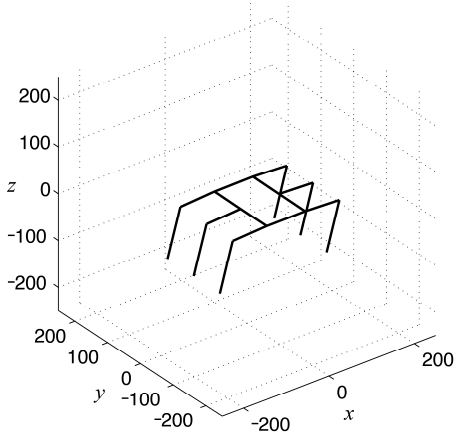
2.13 pav. Sugeneruotos pėdų trajektorijos projekcija *yz* plokštumoje
Fig. 2.13. Generated feet trajectory projection in *yz* plane

Tikrinant kinematinių skaičiavimų teisingumą pirmiausia pavaizduojamas robotas neutralioje stacionarioje būsenoje (2.14 pav.). Kaip matyti iš paveikslėlio, robotas pavaizduojamas be jokių trūkių tarp taškų ir neiškraipytas, forma atitinka reikiamą. Kūno kinematika tikrinama pakeičiant roboto kūno poslinkį ar posūkį apie vieną iš ašių ir pavaizduojant robotą.



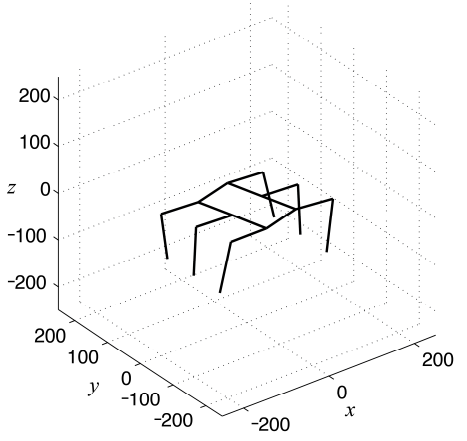
2.14 pav. Neutrali roboto būsena
Fig. 2.14. Neutral robot štance

2.15 paveiksle pateiktas roboto postūmis *x* ašimi 30 mm. Iš paveikslėlio matyti, kad pasikeičia tik roboto kūno padėtis *x* ašimi, kojos užima teisingą padėti, kad perneštų kūną į šią padėtį.



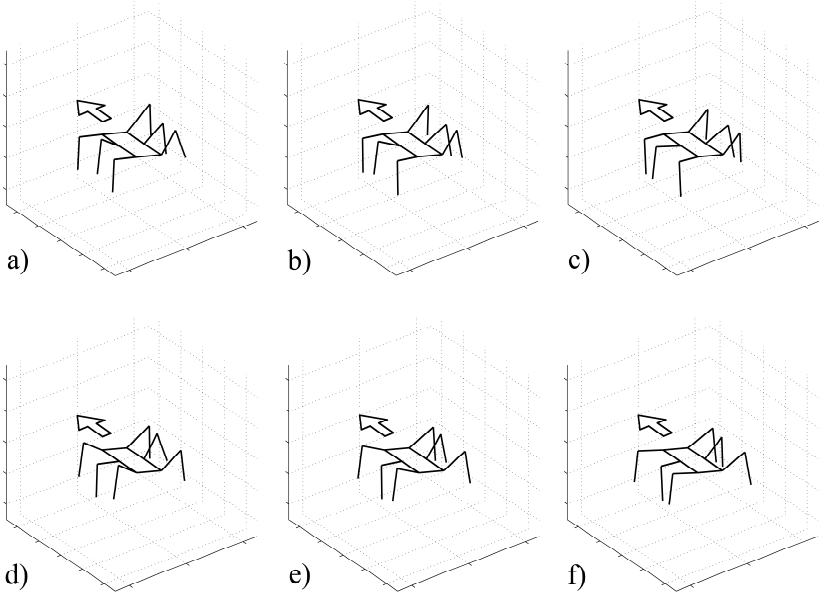
2.15 pav. Roboto kūnas perstumtas *x* ašimi 30 mm
Fig. 2.15. Robot body translation 30 mm along *x* axis

2.16 paveiksle pateiktas roboto posūkis apie *z* ašį 15º. Taip pakeitus roboto pozą, pasikeičia tik kojų padėtys, kad pasuktų kūną reikiama kryptimi, kiti tokie parametrai kaip kūno aukštis ir kiti nesikeičia.



2.16 pav. Roboto kūno posūkis apie *z* ašį 15º
Fig. 2.16. Robot body rotation 15º around *z* axis

Taip pat patikrinami kombinuoti roboto judesiai, tokie kaip pavyzdžiui du skirtingi kūno postūmiai ar posūkiai apie skirtingas ašis, postūmiai ir posūkiai. Patikrinamos visos roboto eisenos esant neutraliai kūno padėčiai bei su pakeista kūno padėtimi. 2.17 paveiksle (a–f) pavaizduotas vienas pilnas roboto žingsnis naudojant trikoję eiseną, kai kūnas perstumtas *z* ašimi –30 mm ir pasuktas apie *y* ašį 15º. Robotui judant kūno padėtis išlieka nepakitusi, o pėdos juda tose pačiose plokštumose.



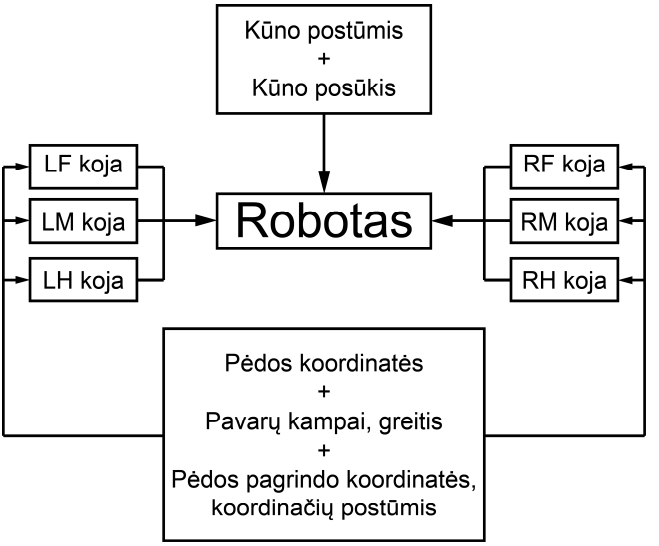
2.17 pav. Roboto trikojės eisenos pilno žingsnio imitavimas
Fig. 2.17. Full step imitation of robot‘s tripod gait

Iš aprašytų roboto imitacinių bandymų matyti, kad roboto kinematinės išraiškos ir pėdų trajektorijų generavimo išraiškos yra teisingos ir gali būti toliau naudojamos fizinio roboto judesių valdymui.

2.2. Roboto valdymo programos sudarymas

Kadangi šešiakojis robotas yra sudėtinga elektromechaninė sistema su daugeliu laisvė laipsnių ir jo judėjimas reikalauja visų galūnių tikslaus sinchroniško judėjimo, norint išgauti sklandžius judesius ar ėjimą, jo valdymo programa gali būti itin sudėtinga. Kuriant roboto valdymo sistemą būtina ją organizuoti taip,

kad roboto valdymas būtų kuo paprastesnis, t. y. jį būtų galima valdyti labai paprastomis komandomis, bet tuo pačiu užtikrintų visišką roboto judesių valdymo laisvę. Teisingas valdymo programos organizavimas ir struktūrizavimas gali ženkliai sumažinti programos sudėtingumą bei palengvinti roboto valdymą (Walas *et al.* 2008; Celaya, Luis Albarral 2003; Berns, K. *et al.* 1999).



2.18 pav. Roboto išskaidymas programos struktūros sudarymui
Fig. 2.18. Robot segments for creating programs structure

Analogiškai kaip imitacinėje programoje, kintamieji sudėti į bendrus masyvus, kad juos būtų patogu naudoti. Robotą galima išskaidyti į kelias pagrindines sudedamąsias dalis (2.18 pav.). Roboto kūną, kurį aprašo jo poza (kūno posūkis ir postūmis) bei šešios kojos, kurias aprašo pėdos koordinatės, pavarų kampai ir greičiai, kojos pagrindo koordinatės ir jų postūmis.

Programoje sukuriamos dvi struktūros, viena aprašanti robotą, kita – kojas. Kojas aprašanti struktūra sudaryta iš 5 masyvų, kurie saugo pėdų koordinates, kojų pagrindų koordinates, pėdų pagrindų koordinačių postūmius, pavarų kampus bei greičius, taip pat iš vieno kojos fazės kintamojo:

```
typedef struct
{
    float foot_coords[3];
    float servo_angles[3];
    float servo_speeds[3];
    float leg_B_coords[3];
    float leg_B_offsets[3];
```

```
    float phase;
} legs;
```

Robotą aprašanti struktūra sudaryta iš masyvo aprašančio kojas, trijų postūmio kintamųjų ir trijų posūkio kintamųjų:

```
typedef struct
{
    legs leg[LEG_NUMBER];

    float x_rotation;
    float y_rotation;
    float z_rotation;

    float x_translation;
    float y_translation;
    float z_translation;
} robot;
```

Robotas aprašomas naudojant `robot hexapod;` eilutę. Naudojant tokį programos kintamųjų struktūrizavimą, kreipimasis į pagrindinius roboto kintamuosius tampa labai patogus ir lengvai suprantamas. Taip pat visi kintamieji yra saugomi vienoje globalioje struktūroje, ir tai leidžia į juos kreiptis iš bet kurios programos vietos. Pavyzdžiui, norint pakeisti roboto kūno posūkį apie *x* ašį, bet kurioje programos vietoje užtenka tokios eilutės:

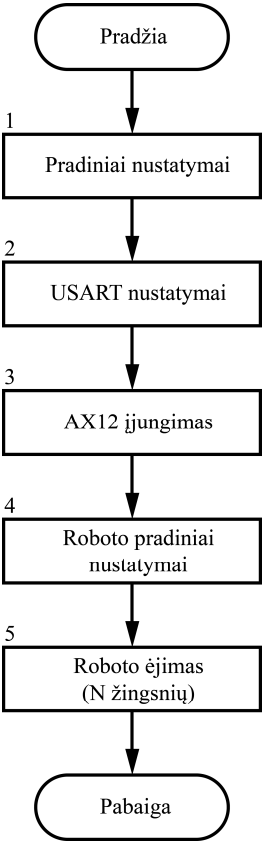
```
hexapod.x_rotation = 15;
```

Kreipimasis į roboto kojų kintamuosius yra analogiškas, pavyzdžiui vidurinės kairės kojos *y* koordinatės pakeitimas atrodo taip:

```
hexapod.leg[LEFT_MIDDLE].foot_coords[y]=30;
```

Kaip ir bet kuri programa, roboto valdymo programa prasideda nuo pradinių nustatymų (2.19 pav., 1). Šiuo atveju tai būtų mikrovaldiklio įvesties ir išvesties prievadų nustatymas, laikmačių konfigūravimas bei paleidimas, I2C sąsajos ir akcelerometro nustatymai. Nuoseklioji sąsaja (USART) konfigūruojama pagal AX-12 pavarų sąsajos reikalavimus ir nustatoma 1 Mbps duomenų perdavimo greičiui, 8 bitų duomenų paketui su 1 stop bitu (2.19 pav., 2). Kadangi mikrovaldiklis dirba 16 MHz taktų dažniu, tai naudojant 1 Mbps USART mainų greitį gaunama 0 % klaidų tikimybė perduodant duomenis. Tada įjungiamos visos roboto pavaros su funkcija AX12_TORQUE_ENABLE() (2.19 pav., 3), kurią įvykdžius pavaros pereina į padėties išlaikymo režimą ir priešinasi išoriniams momento poveikiams. Sekantis žingsnis yra roboto kintamųjų inicializacija (2.19 pav., 4). Iššaukiama funkcija CONFIG(), kuri įrašo į roboto kūno poslinkio ir posūkio kintamuosius pradines vertes, pėdų koordinatėms priskiriamos pradinės vertės lygios 0, nustatomi pavarų greičiai, naudojamas maksimalus 150 aps./min greitis, įrašomi pėdų pagrindų koordinačių postūmiai. Visos į kintamuosius įrašomos vertės nustato roboto pradinę būseną, kurioje jis pradės ėjimą. Keičiant

šias vertes galima nustatyti kitokią pradinę roboto būseną. Nustačius roboto pradinę būseną, vykdoma roboto ėjimo funkcija MOVE(*step_no*, *dir*, *gait*, *mode*) (2.19 pav., 5). Ši funkcija kartojama tiek kartų kiek, nurodo kintamasis *step_no*.



2.19 pav. Roboto valdymo programos algoritmas
Fig. 2.19. Algorithm of robot‘s control program

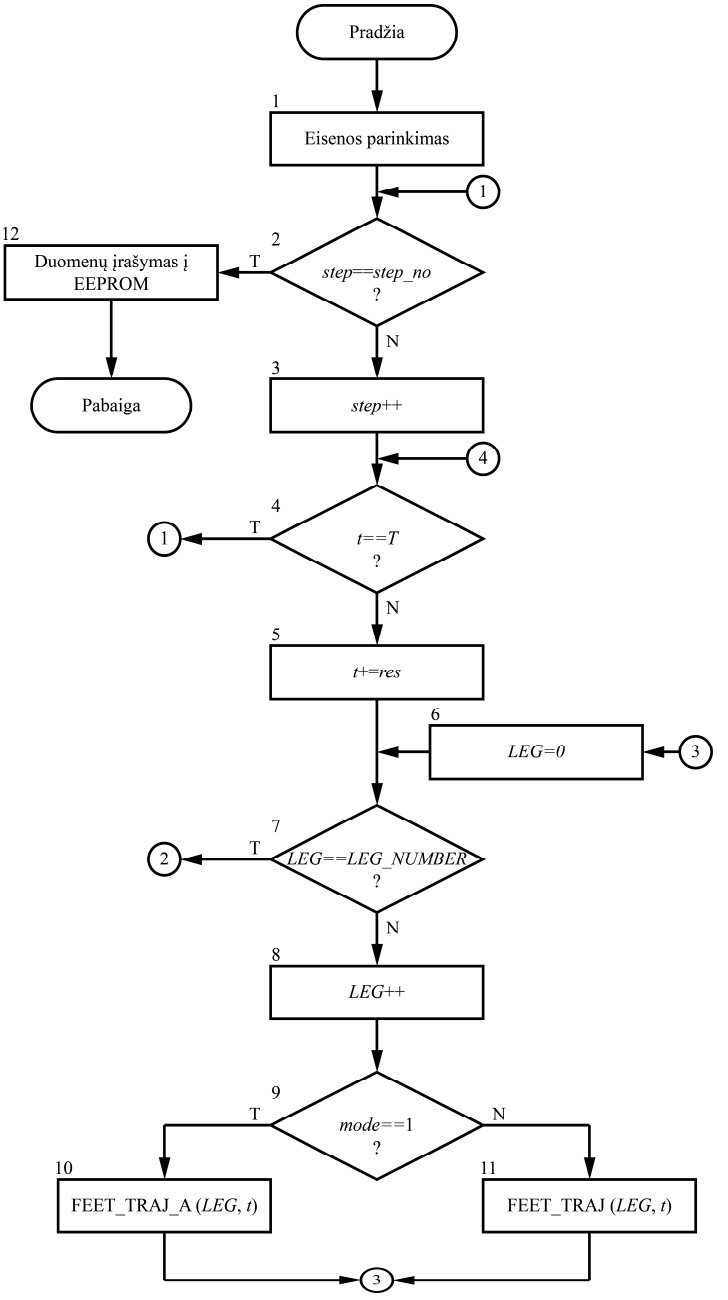
Pagrindinė visos programos funkcija, kurioje atliekamas pagrindinis roboto valdymas, yra MOVE(). Todėl šios funkcijos algoritmas pateikiamas atskirai (2.20–2.22 pav.). Iššaukiant funkciją MOVE() reikia nurodyti keturis argumentus: žingsnių skaičių *step_no*, roboto ėjimo kryptį *dir*, roboto eiseną *gait* ir ėjimo režimą *mode*. Pirmiausia funkcijoje yra iššaukiama roboto eisenos parinkimo funkcija GAIT(), kuri pagal argumentą *gait* parenką eisenos parametrus (2.20 pav., 1). Eisenos parametrų parinkimas yra analogiškas roboto imitacinėje programoje panaudotam. Toliau programa patenka į pagrindinį MOVE() funkcijos

ciklą (2.20 pav., 2 ir 3), kuris yra kartojamas tiek kartų, kiek nurodyta *step_no* argumente. Prieš įeinant į ciklą, programa patikrina, ar einamas žingsnis *step* yra lygus žingsnių skaičiui *step_no*. Jei ne, programa didina kintamąjį *step* vienetu ir įeina į ciklą, jei taip – ciklas yra baigiamas, išsiunčiami duomenys į išorinę pastoviąją duomenų atmintį (EEPROM), vėlesniam duomenų apdorojimui ir analizei, (2.20 pav. 12) ir funkcija MOVE() baigiama vykdyti.

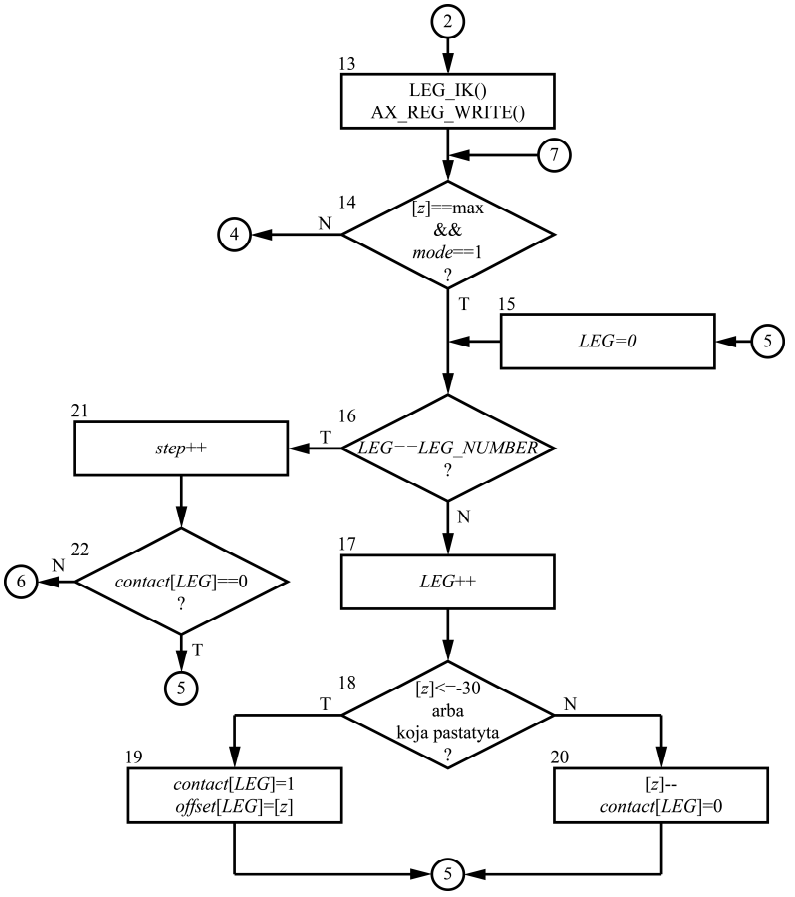
Žingsnių cikle iškart pereinama prie laiko tarpsnių ciklo (2.20 pav., 4 ir 5), kuriame tikrinama ar laiko kintamasis *t* yra lygus eisenos periodui *T*. Laiko tarpsnis (kintamojo *t* didinimo periodas) priklauso nuo šios funkcijos vykdymo laiko arba galimas fiksuoto ilgio laiko tarpsnis naudojant valdiklio laikmačio pertrauktis. Jei *t* nelygus $T - t$ kintamasis didinamas žingsniu *res* ir įeinama į ciklą, jei lygus – programa išeina iš laiko tarpsnių ciklo ir pereina prie išorinio žingsnių ciklo. Laiko tarpsnio cikle priklausomai nuo ėjimo režimo *mode* yra parenkama pėdų koordinačių generavimo funkcija, jei *mode* = 1 parenkama adaptyvaus ėjimo koordinačių generavimo funkcija (2.20 pav., 10), jei *mode* = 0 – neadaptyvaus ėjimo funkcija (2.20 pav., 11). Abiem atvejais į funkcijas yra kreipiamasi šešis kartus (kiekvienai kojai atskirai), tai atliekama ciklu (2.20 pav., 7 ir 8).

Suskaičiavus visų pėdų koordinates einamu laiku, suskaičiuojami ir išsiunčiami reikiami kampai pavaroms (2.21 pav., 13). Po kiekvieno koordinačių perskaičiavimo ir išvedimo tikrinama, ar bet vienos pėdos koordinatės pasiekė maksimalią vertę ir ar kintamasis *mode* lygus vienam (2.21 pav., 14). Jei nė viena sąlyga netenkinama, programa grįžta į laiko tarpsnio ciklo pradžią. Jei tenkinamos abi sąlygos, tada programa mažais žingsniais nuleidinėja pakeltas kojas (2.21 pav., 16–20). Jei pėdos *z* koordinatė tampa lygi maksimaliam galimos duobės gyliui (pasirinkta vertė lygi –30 mm) arba koja pastatoma ant paviršiaus, tos kojos programa daugiau nebeleidžia žemyn, uždeda vėliavėlę, kad atitinkama koja pasiekė paviršių (*contact*[*LEG*] = 1) ir išsaugomas žingsnio aukščio pokytis (*offset*[*LEG*] = *z*) (2.21 pav., 19). Visa tai kartojama tol, kol bent viena koja yra pakelta (2.21 pav., 22).

Kai tik visų kojų vėliavėlės yra nustatytos (lygios 1) išeinama iš esamo ciklo ir dar patikrinama ar visos kojos pastatytos ant paviršiaus (ar visų kojų jungikliai paspausti) ir *mode* lygus 1 (2.22 pav., 23). Jei ši sąlyga tenkinama, programa suskaičiuoja pėdų nuokrypas, atsižvelgiant į roboto kūno pokrypį ir nusėdimą (2.22 pav., 24). Kiekvienos pėdos *z* koordinatė yra pakoreguojama pagal apskaičiuotas nuokrypas (2.22 pav., 25), suskaičiuojamas pėdų *z* koordinačių vidurkis, pakoreguojamas, jei reikia, kūno poslinkis *z* ašimi, suskaičiuojami ir išsiunčiami kampai pavaroms bei išsaugomi nauji žingsnio aukščio pokyčiai (2.22 pav., 26–28). Programa toliau tęsiama vėl tikrinant ar bent vienos iš pėdų *z* koordinatė pasiekė maksimalią vertę.



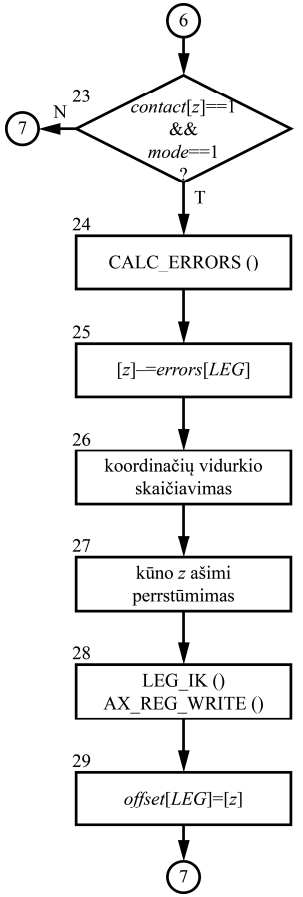
2.20 pav. Funkcijos MOVE() algoritmas
Fig. 2.20. Algorithm of MOVE() function



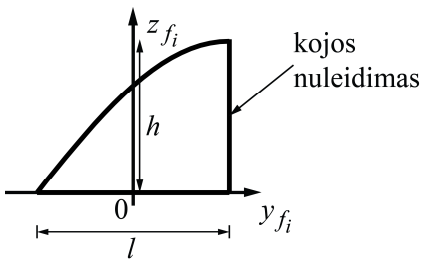
2.21 pav. Funkcijos MOVE() algoritmo tęsinys
Fig. 2.21. Continuation of MOVE() function algorithm

Kintamasis *mode* leidžia pasirinkti roboto ėjimo režimą, kuris parenka atitinkamas pėdų trajektorijų generavimo funkcijas. Jei parenkamas neadaptyvus režimas, naudojama FEET_TRAJ() funkcija, kuri naudoja (2.24)–(2.26) išraiškas trajektorijų skaičiavimams. Tačiau, ši funkcija neleidžia robotui prisitaikyti prie paviršiaus todėl adaptyviam režimui naudojam modifikuota funkcija FEET_TRAJ_A(). Šios funkcijos generuojamos trajektorijos projekcija *yz* ašyse pavaizduota 2.23 paveiksle.

2.23 paveiksle pavaizduotas kojos nuleidimas atliekamas taikant algoritmą pavaizduotą 2.21 paveiksle, 17–20.



2.22 pav. Funkcijos MOVE() algoritmo tęsinys
Fig. 2.22. Continuation of MOVE() function algorithm



2.23 pav. Adaptyvaus ėjimo pėdų trajektorijos projekcija *yz* plokštumoje
Fig. 2.23. Feet trajectory for adaptive locomotion projection in *yz* plane

2.3. Paviršiaus vertinimo ir sprendimų priėmimo metodų aprašymas

Paviršiaus nelygumo įvertinimui naudojama pėdų *z* koordinačių standartinė nuokrypa σ. Ji nusako taškų išsibarstymą apie jų bendrą vidurkį. Lyginant su kitais paviršiaus nelygumo nustatymo metodais, tokiais kaip regos sistemos, σ parametras yra labai lengvai apskaičiuojamas. Dažniausiai σ parametras yra skaičiuojamas jau žinant viso vertinamo paviršiaus taškų koordinates. Siūlomas paviršiaus nelygumo vertinimo metodas remiasi tuo, kad σ parametras skaičiuojamas tik tam tikru momentu ir tik tam paviršiaus plotui, kuriame tuo momentu yra robotas. Tokiu būdu σ parametras parodo paviršiaus nelygumą roboto pėdomis apribotame plote.

Paviršiaus standartinė nuokrypa σ skaičiuojama pagal roboto pėdų *z* koordinates tik tada, kai visos kojos yra pastatytos ant paviršiaus. Pėdų koordinatės paimamos iš pėdų koordinačių sistemos $x_{f_i} y_{f_i} z_{f_i}$, kurios roboto koordinačių sistemoje $X_0 Y_0 Z_0$ yra transformuotos taip, kad robotui stovint neutralioje padėtyje (2.24 pav., a), pėdų *z* koordinatės yra lygios 0. Taigi *z* koordinatės atitinka pėdų aukščio išsibarstymą nuo neutralios padėties. Koordinatė *z* yra teigiama, jei koja yra pakeliama, ir neigiama, jei nuleidžiama. Kai $z_{RF} = z_{RM} = z_{RH} = z_{LF} = z_{LM} = z_{LH}$, visos pėdos yra vienoje, idealiai lygioje, plokštumoje. Paviršius vertinamas po kiekvieno žingsnio, todėl yra žinoma ant kokio paviršiaus robotas stovi tuo momentu. Priimama, kad roboto kūnas visą laiką išlieka horizontalus (tikrinama vertinant akcelerometro duomenis). Standartinės nuokrypos σ vertė skaičiuojama pagal (2.27) formulę.

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(z_f(i) - \overline{z}\right)^2}, \tag{2.27}$$

$$\overline{z} = \frac{\sum_{i=1}^{n} z_f(i)}{n}, \tag{2.28}$$

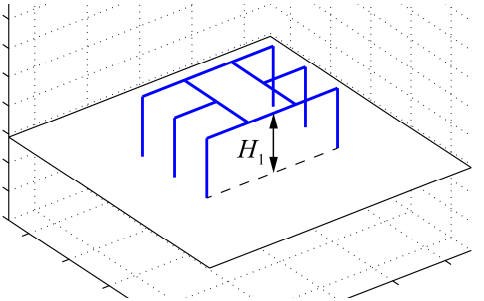
čia *n* – kojų skaičius (šiuo atveju lygus 6), $z_f(i)$ – *i*-osios kojos koordinatė.

Standartinės nuokrypos σ skaičiavimo sudėtingumą sumažina ir tai, kad pėdų koordinatės yra žinomos visos roboto ėjimo fazės metu, todėl nereikia papildomai skaičiuoti pėdų koordinačių. Neigiamas tokio paviršiaus vertinimo metodo aspektas yra mažas taškų skaičius, kas gerokai sumažina paviršiaus nelygumo

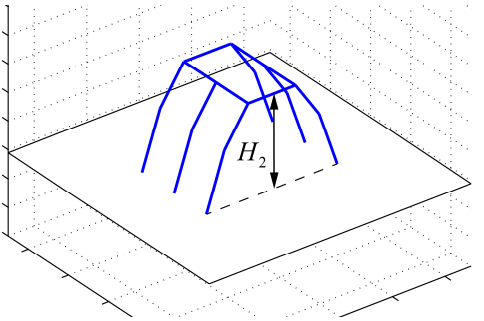
vertinimo tikslumą, ir dviejų gretimų žingsnių σ vertės gali ženkliai skirtis. Tačiau yra žinoma, kad didėjant paviršiaus nelygumui, σ vertės didėja, todėl didesnė σ vertė atitiks didesnius paviršiaus nelygumus.

2.3.1. Paviršiaus nelygumų ribinės vertės

Tiriamo šešiakojo roboto praeinamumas priklauso nuo roboto kojų matmenų. Neiškėlus kūno roboto praeinamumas yra lygus kojos blazdos ilgiui l_2 ir lygus 87 mm (2.24 pav., a) Praeinamumas lygus 160 mm, jei roboto kūnas yra iškeltas maksimaliai (2.24 pav., b).



a)



b)

2.24 pav. Roboto stovėjimo būsenos ir praeinamumų aukščiai. a) neutrali stovėjimo būsena ir jos praeinamumo aukštis H_1, b) stovėjimo būsena su maksimaliai iškeltu kūnu ir jos praeinamumo aukštis H_2

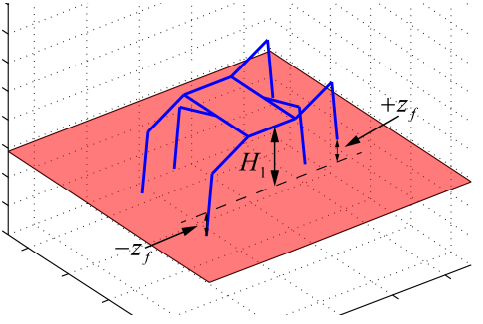
Fig. 2.24. Robot stance and their clearance heights. a) neutral štance and its clearance height H_1, b) stance with maximum lifted body and its clearance height H_2

Kūno iškėlimo aukštį nulemia kojų blauzdos ir šlaunies ilgiai l_2 ir l_1. Remiantis tuo galima teigti, kad paviršius, kurio nelygumai yra 160 mm ir daugiau yra nepraeinamas paviršius. Žinant maksimalų praeinamo paviršiaus nelygumų aukštį, paviršiaus nelygumus galima sugrupuoti į tokias grupes:

- didelis nelygumas: 87 mm $\leq h <$ 160 mm;
- vidutinis nelygumas: 43 mm $\leq h <$ 87 mm;
- mažas nelygumas: 0 mm $< h <$ 43 mm;
- lygus paviršius: h = 0 mm.

Didelio nelygumo grupei priskiriami visi nelygumai kurių aukščiai yra mažesni už maksimalų roboto praeinamumą iškėlus kūną ir mažesni ar lygus maksimaliam praeinamumui neiškėlus kūno. Vidutinio nelygumo grupei priskiriami nelygumai, kurių aukščiai mažesni už maksimalų praeinamumą neiškėlus kūno ir didesni ar lygus pusei šio aukščio. Mažo nelygumo grupei priskiriami nelygumai mažesni už pusę maksimalaus praeinamumo neiškėlus kūno.

Norint įvertinti nežinomo paviršiaus σ vertes, pagal išskirtus paviršiaus grupių aukščius galima apskaičiuoti jų σ vertes. Maksimali σ vertė bus pasiekta tuo atveju, kai trys roboto kojos bus iškeltos į teigiamą maksimalų aukštį, o kitos trys – į neigiamą. Pavyzdžiui, kraštinė σ vertė mažo nelygumo paviršiui bus pasiekta, kai trijų pėdų z koordinatės bus lygios +43 mm, o kitų trijų –43 mm (2.25 pav.). Taigi, paviršių nelygumai taip pat gali būti sugrupuoti pagal jų σ vertes, kurios sutampa su aukščių vertėm.



2.25 pav. Roboto pėdų padėtys esant mažo nelygumo paviršiui
Fig. 2.25. Robot feet positions in low roughness terrain

Daugeliu atveju, robotui einant natūraliu paviršiumi, σ vertės (ir jų vidurkis) bus gerokai mažesnės, nei apskaičiuotos. Pavyzdžiui tam, kad σ vertės visą laiką išliktų didelio nelygumo paviršiaus diapazone, roboto pėdų koordinatės trims kojoms turi būti $+h$, kitoms trims kojoms $–h$, kur h priklauso didelio nelygumo aukščių diapazonui. Tačiau, netgi jei paviršiaus nelygumai pastoviai pri-

klausytų didelio nelygumo aukščių diapazonui, yra tikimybė, kad kelios ar net visos pėdos bus pastatomos į tokį patį ar panašų aukštį, ir σ vertė bus labai maža ar artima 0.

2.3.2. Paviršiaus nelygumo nustatymas pagal kampus tarp plokštumų

Žinant roboto pėdų aukščio koordinates atsiranda galimybė įvertinti ne tik paviršiaus nelygumą, bet ir paviršiaus šlaito gradientą. Paviršiaus gradientą (Zhang, Zheng 2008) galima nustatyti skaičiuojant kampus tarp esminių roboto plokštumų. Kadangi plokštumą galima aprašyti trimis taškais, tai per šešias roboto pėdas galima išvesti dvi plokštumas (2.26 pav.).

Plokštuma išvesta per LF-LH-RM pėdas yra vadinama pirmąją ($n = 1$), plokštuma per RF-RH-LM pėdas – antrąją ($n = 2$), n – plokštumos numeris. Plokštumas aprašo tokia lygčių sistema (2.29).

$$\begin{cases} a_n x_{f_i} + b_n y_{f_i} + c_n z_{f_i} + d_n = 0, \\ a_n x_{f_j} + b_n y_{f_j} + c_n z_{f_j} + d_n = 0, \\ a_n x_{f_k} + b_n y_{f_k} + c_n z_{f_k} + d_n = 0, \end{cases} \tag{2.29}$$

čia

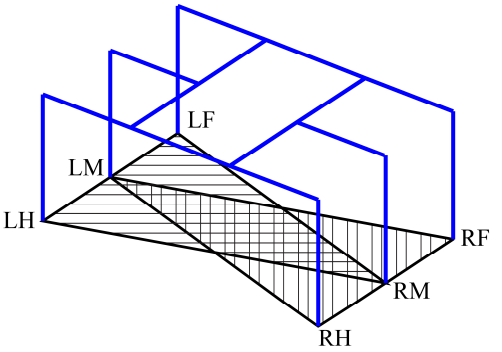
$$a_n = \frac{-d_n}{D_n} \begin{vmatrix} 1 & y_{f_i} & z_{f_i} \\ 1 & y_{f_j} & z_{f_j} \\ 1 & y_{f_k} & z_{f_k} \end{vmatrix}, \tag{2.30}$$

$$b_n = \frac{-d_n}{D_n} \begin{vmatrix} x_{f_i} & 1 & z_{f_i} \\ x_{f_j} & 1 & z_{f_j} \\ x_{f_k} & 1 & z_{f_k} \end{vmatrix}, \tag{2.31}$$

$$c_n = \frac{-d_n}{D_n} \begin{vmatrix} x_{f_i} & y_{f_i} & 1 \\ x_{f_j} & y_{f_j} & 1 \\ x_{f_k} & y_{f_k} & 1 \end{vmatrix}, \tag{2.32}$$

$$D_n = \begin{vmatrix} x_{f_i} & y_{f_i} & z_{f_i} \\ x_{f_j} & y_{f_j} & z_{f_j} \\ x_{f_k} & y_{f_k} & z_{f_k} \end{vmatrix}, \tag{2.33}$$

n – plokštumos numeris, i, j, k – kojų indeksai.



2.26 pav. Plokštumos naudojamos paviršiaus gradiento skaičiavimui
Fig. 2.26. Planes that are used to calculate terrain gradient

Apskaičiavus visus reikiamus parametrus (2.29) lygčiai pagal (2.30)–(2.33) lygtis, plokštumų išraiškas galima užrašyti įprastine forma (2.34).

$$f(a,b,c) = ax + by + cz + d = 0\,. \tag{2.34}$$

Plokštumų posvyrio kampas su horizontalios plokštumos statmeniu, gali būti apskaičiuojamas pagal (2.35) ir (2.36) formules.

$$\alpha_1 = -\operatorname{atan}\frac{a_1}{c_1}, \alpha_2 = -\operatorname{atan}\frac{a_2}{c_2}, \tag{2.35}$$

$$\beta_1 = -\operatorname{atan}\frac{b_1}{c_1}, \beta_2 = -\operatorname{atan}\frac{b_2}{c_2}, \tag{2.36}$$

čia α_1, α_2 – pirmosios ir antrosios plokštumų pokrypio kampai x ašimi, β_1, β_2 – pirmosios ir antrosios plokštumų pokrypio kampai y ašimi. Jei kampai α_1 ir α_2 yra teigiami, šlaitas yra robotui iš dešinės, jei neigiami – iš kairės. Jei kampai β_1

ir β_2 yra teigiami, robotas lipa į kalną, jei neigiamas – leidžiasi. Šie kampai kartu su vertikaliu pėdų poslinkiu, c_1 ir c_2, taip pat leidžia įvertinti paviršiaus nelygumus. Jei $\alpha_1 = \alpha_2 = 0$, $\beta_1 = \beta_2 = 0$ ir $c_1 = c_2$ reiškia, kad robotas stovi ant ideliai lygaus paviršiaus. Jei $\alpha_1 = \alpha_2 \neq 0$ ir (arba) $\beta_1 = \beta_2 \neq 0$ ir $c_1 = c_2$, robotas stovi ant lygaus šlaito. Jeigu $\alpha_1 \neq \alpha_2$ ir (arba) $\beta_1 \neq \beta_2$ ir (arba) $c_1 \neq c_2$ roboto pėdos yra ne vienoje plokštumoje ir reiškia, kad paviršius, kuriuo juda robotas yra nelygus.

2.3.3. Sprendimų priėmimas

Roboto judėjimas skirtingo nelygumo paviršiais skiriasi. Einant lygiu paviršiumi robotas gali parinkti greitesnę, bet mažiau stabilią eiseną, tačiau einant labai nelygiu paviršiumi gali reikti eiti lėtai, bet apgalvotai ir atsargiai. Taip pat, priklausomai nuo užduoties, robotui gali būti reikalavimas tikslą pasiekti kaip įmanoma greičiau (kritinės situacijos), išlaikyti kūno stabilumą arba tam tikrą padėtį (krovinių gabenimas) arba netgi judėti atsargiai (nuotoliniai aplinkos tyrimai). Roboto greičiui, stabilumui ir atsargumui gali būti suteikti prioritetai, norint suteikti vienam iš kriterijų didesnę svarbą. Atsižvelgiant į visa tai, sudaromos tokios taisyklės roboto ėjimui:

- Ėjimui lygiu paviršiumi, turi būti parenkama greičiausia eisena.
- Didėjant paviršiaus nelygumui, turi būti parenkama lėtesnė, bet stabilesnė eisena.
- Kai paviršiaus nelygumas tampa didelis, parenkama atsargiausia eisena.
- Šios taisyklės turi leisti lengvai (keičiant atitinkamus koeficientus) keisti roboto elgseną esant skirtingiems paviršiams.

Kiekviena eisena yra reitinguojama skalėje nuo 1,0 (labai gera) iki 0,0 (labai bloga). Eisenos reitingas parodo jos gebėjimą užtikrinti greitį, stabilumą arba atsargumą. Kiekvienam kriterijui suteikiamas svorio koeficientas, kuris nurodo kriterijaus svarbą. Koeficientų skalė yra nuo 1,0 (labai svarbu) iki 0,0 (nesvarbu). Šie koeficientai nulemia eisenų reitingų kitimo greitį. Priimama, kad roboto greitis ir stabilumas yra vienodai svarbūs ($w_1 = w_2 = 0{,}3$), bet atsargumas svarbesnis ($w_3 = 0{,}4$). Svorių koeficientų suma turėtų būti lygi 1.

Keičiami turėtų būti tik šie prioritetai, norint išgauti skirtingas roboto elgsenas. Prioritetų skalė gali būti laisvai pasirenkama, kuo platesnė skalė ir skirtumai tarp prioritetų verčių, tuo roboto valdymo sistema bus mažiau jautri nelygumų pokyčiams. Pavyzdžiui, jei roboto užduotis yra saugiai nugabenti krovinį iki tikslo, tai stabilumui turi būti skirtas aukščiausias prioritetas, kiek mažesnis atsargumui, greičio prioritetas turėtų būti gerokai mažesnis už stabilumo ar atsar-

gumo kriterijų prioritetus. Tai lems, kad roboto sistema stengsis parinkti tokią eiseną, kuri geriausiai užtikrina roboto stabilumą ir atsargumą.

Klasikinė sprendimų matrica yra statinė, t. y. tinkama tik vieninteliam atvejui. Tačiau paviršiaus nelygumas keičiasi, ir roboto valdymo sistema turi parinkti tinkamiausią eiseną esamam nelygumui. Todėl sprendimų matricoje turi būti atsižvelgta į σ parametrą. Kiekvienos eisenos reitingas turėtų keistis, keičiantis σ vertei, o eisenų svorio koeficientai ir prioritetai nulemti eisenos reitingo kitimo greitį ir pradinė vertę (Ostrowski *et al.* 2000; Silva *et al.* 2004; Wang 2005; Mitsunaga, Asada 2006; Umm-e-Habiba, Asghar 2009).

2.2 lentelė. Sprendimų matrica su kintamais parametrais skirta roboto eisenos parinkimui
Table 2.2. Decision matrix with varying parameters used to select robot gait

Kriterijus	Prioritetas	Svoris	Trikojė	Keturkojė	Banguojanti	Pulsuojanti
Greitis	p_1	w_1	1	0,7	0,4	0,4
Stabilumas	p_2	w_2	0,5	0,7	0,8	0,9
Atsargumas	p_3	w_3	0,2	0,5	0,9	1
			r_1	r_2	r_3	r_4

$$r_1 = \sigma \cdot (1 \cdot w_1 + 0,5 w_2 + 0,2 w_3) + p_1 \tag{2.37}$$

$$r_2 = \sigma \cdot (0,7 \cdot w_1 + 0,7 w_2 + 0,5 w_3) + (p_1 + p_2)/2, \tag{2.38}$$

$$r_3 = \sigma \cdot (0,4 \cdot w_1 + 0,8 w_2 + 0,9 w_3) + (p_2 + p_3)/2, \tag{2.39}$$

$$r_4 = \sigma \cdot (0,4 \cdot w_1 + 0,9 w_2 + 1 w_3) + p_3. \tag{2.40}$$

2.2 lentelė parodo eisenoms priskirtus reitingus. Šie reitingai parodo eisenos gebėjimą užtikrinti roboto greitį, stabilumą ar atsargumą. $r_1 - r_4$ yra kiekvienos eisenos reitingų suminės vertės, kurios skaičiuojamos pagal (2.37)–(2.40) išraiškas.

2.4. Antrojo skyriaus išvados

1. Šešiakojo žingsniuojančio roboto kinematinis modelis sudarytas taikant homogenines transformacijų matricų bei atvirkštinės kinematikos vienai kojai metodus. Toks kinematinių skaičiavimų išskaidymas supaprastina roboto kinematinio modelio sudarymą.
2. Roboto imitavimas MATLAB® aplinkoje leidžia išbandyti kinematinius skaičiavimus prieš įrašant juos į roboto valdymo sistemą, tai leidžia išvengti roboto gedimų susijusių su skaičiavimų netikslumais. Taip pat, roboto imitavimas leidžia patikrinti skirtingus valdymo ir roboto judėjimo algoritmus.
3. Sudarytos roboto pėdų trajektorijų generavimo matematinės išraiškos visoms šešiakojo roboto eisenoms, keičiant kiekvienos kojos fazę ir viso žingsnio periodą. Šis algoritmas užtikrina paprastą roboto žingsnio ilgio, aukščio bei ėjimo krypties keitimą.
4. Sudarytas supaprastintas šešiakojo roboto valdymo algoritmas leidžia robotą valdyti paprastomis aukšto lygio komandomis. Algoritmas leidžia keisti roboto kūno padėtį bei eisenos parametrus, robotui einant.
5. Roboto pėdų *z* koordinačių standartinė nuokrypa leidžia vienu parametru kiekybiškai įvertinti paviršiaus nelygumą. Tai supaprastina paviršiaus nelygumo atpažinimą, kadangi pėdų koordinatės yra visada žinomos. Paviršiaus gradientą galima vertinti skaičiuojant kampus tarp dviejų roboto pėdų plokštumų bei tarp kampo tarp šių plokštumų ir roboto kūno statmens.
6. Parinktos slenkstinės *z* koordinatės standartinės nuokrypos vertės keturioms paviršiaus nelygumo kategorijoms leidžia šias vertes pritaikyti roboto eisenos parinkimui.
7. Pasiūlytas naujas paviršiaus nelygumo ir pokrypio įvertinimo metodas nubrėžiant dvi plokštumas per roboto pėdas. Apskaičiavus kampus tarp vertikalės, nustatytos pagal akcelerometro duomenis, ir šių plokštumų, kampų skirtumas įvardina paviršiau nelygumą, o vidurkis – posvyrį.
8. Taikant sprendimų matricą su kintamais parametrais galima atsiriboti nuo paviršiaus nelygumų skirstymo į kategorijas, kas leidžia roboto valdymo sistemai tiesiogiai reaguoti į paviršiaus nelygumo kitimą, pavyzdžiui keičiant eiseną.

3

Imitaciniai ir eksperimentiniai šešiakojo roboto judėjimo tyrimai

Šio tyrimo tikslas yra ištirti roboto judėjimą nelygiu paviršiumi ir patikrinti paviršiaus vertinimo metodą, įvertinti paviršiaus vertinimo paklaidas ir patikrinti eisenos parinkimo algoritmą.

Šiame skyriuje pateikiamas paviršiaus standartinės nuokrypos σ verčių nustatymas imituojant roboto ėjimą skirtingais paviršiais. Iš anksto žinant paviršių σ vertes galima nusakyti realaus paviršiaus nelygumą. Taip pat aptariamas roboto judėjimas lygiu paviršiumi, taikant adaptyvią eiseną bei paklaidų kompensavimo tyrimas. Aprašomi roboto judėjimo ir paviršiaus σ nustatymo tyrimo rezultatai. Taip pat pateikiami roboto sprendimų priėmimo sistemos imitavimo rezultatai. Sistema leidžia parinkti roboto eiseną, priklausomai nuo paviršiaus nelygumo (kuris išreiškiamas kaip standartinė nuokrypa σ) ir kitų roboto užduočiai keliamų reikalavimų.

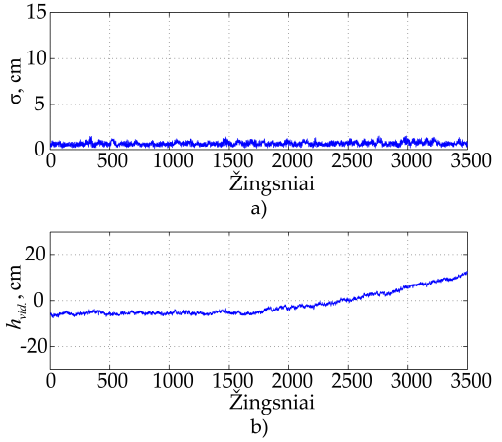
Skyriaus tematika paskelbti trys autoriaus straipsniai (Luneckas 2011; Luneckas 2012; Luneckas, Udris 2013).

3.1. Roboto ėjimo skirtingais paviršiais imitavimas

3.1.1. Standartinės nuokrypos verčių nustatymas

Norint numatyti kaip keisis σ vertės, robotui einant nelygiu paviršiumi, atliekama simuliacija imituojant roboto ėjimą ant sugeneruoto nelygaus paviršiaus. Paviršius yra generuojamas naudojant modifikuotą fraktalinio paviršiaus generavimo programą (Lakaemper 2004), kuri paviršių generuoja taikant *diamond square* algoritmą. Programa yra modifikuota taip, kad būtų galima tiksliai keisti paviršiaus nelygumų aukštį. Nelygumų aukščių vertės paimamos iš ankstesniame skyriuje apibrėžtų kategorijų. Įvedamos nelygumų aukščių vertės nulemia aukščių ribines vertes.

Visiškai ar beveik visiškai lygaus paviršiaus σ vertės yra lygios arba beveik lygios 0, todėl neverta tokio paviršiaus imituoti. Pirmiausia buvo imituotas ėjimas mažo nelygumo paviršiumi (h = ±2 cm), antra imitacija atlikta su didelio nelygumo paviršiumi (h = ±16 cm). Tarpinių variantų imitacija nelabai turi prasmės, nes jų rezultatai pateks į kraštutinių imitacijų rezultatų ribas.

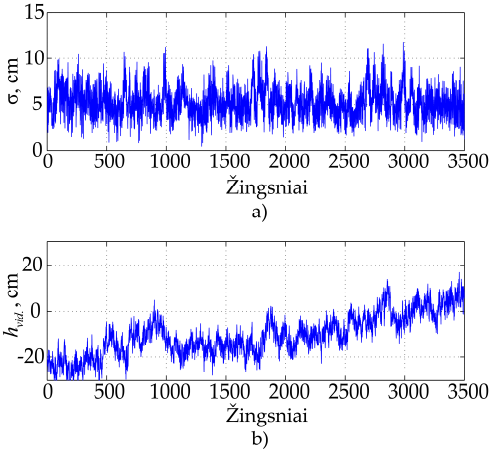


3.1 pav. Standartinės nuokrypos vertės, imituojant roboto ėjimą mažo nelygumo paviršiumi: a) standartinės nuokrypos profilis, b) paviršiaus profilis

Fig. 3.1. Standard deviation values imitating robot locomotion on low roughness terrain: a) Standard deviation profile, b) terrain profile

Paviršius yra generuojamas kaip aukščių matrica, kurios kiekvieno elemento vertė yra paviršiaus aukštis tame taške. Roboto judėjimas tokiu paviršiumi yra

imituojamas priskiriant atitinkamas aukščių koordinates atitinkamų pėdų *z* koordinatėms. Vaizdžiai tai galima paaiškinti, kaip roboto perstatymą į naują vietą vienodais žingsniais.



3.2 pav. Standartinės nuokrypos vertės, imituojant roboto ėjimą didelio nelygumo paviršiumi: a) standartinės nuokrypos profilis, b) paviršiaus profilis

Fig. 3.2. Standard deviation values imitating robot locomotion on high roughness terrain: a) Standard deviation profile, b) terrain profile

Iš 3.1 ir 3.2 paveikslų aiškiai matyti, kad σ vertės (σ vidurkis) nepriklauso nuo to, ar robotas eina šlaitu. σ vertės daugiausia priklauso nuo lokalaus paviršiaus nelygumo. Taip pat, pabrėžtina, kad didėjant paviršiaus nelygumui didėja ne tik σ vidurkis, bet taip pat didėja ir σ verčių ribos, t. y. kuo didesnis paviršiaus nelygumas, tuo σ vidurkis didesnis ir vertės išsibarstę didesniame diapazone, dviejų gretimų žingsnių σ vertės gali ženkliai skirtis. Kadangi 3.1 ir 3.2 paveiksluose pateiktos σ vertės gali būti laikomos ribinėmis, tai robotui judant natūraliu paviršiumi, gaunamos jo σ vertės bus šiose ribose. Yra tikimybė, kad susidarys tokia situacija, kai didelio nelygumo paviršiaus σ vertės pateks į mažesnio nelygumo intervalą. Tai parodo, kad netgi vertės iš dviejų kraštinių intervalų gali persidengti, o vertės iš artimesnių intervalų persidengs žymiai daugiau. Šios priežastys sudaro sunkumų skirstant σ į griežtas kategorijas.

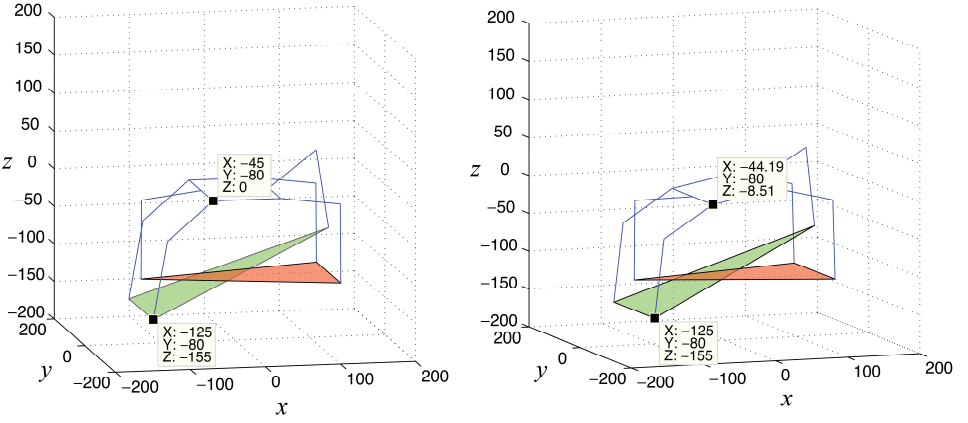
Be to, paviršiaus nelygumas gali būti ne vienintelis kriterijus įtakojantis roboto eisenos parinkimą. Reikalavimai roboto atliekamai užduočiai gali nustatyti kitus, ne mažiau svarbius, kriterijus eisenos parinkimui. Pavyzdžiu gali būti gyvūnas, kurį vejasi grobuonis, tokiu atveju auka pasirinks greitesnę eiseną nepri-

klausomai nuo to kokiu paviršiumi ji bėga. Remiantis tuo, priimama, kad σ vertė (arba paviršiaus nelygumas) turi įtakoti, bet nenulemti roboto eisenos parinkimo.

3.1.2. Paviršiaus pokrypio nustatymo imitavimas

Skirtingos roboto būsenos ant paviršiaus sukuriamos keičiant roboto pėdų *z* koordinates. Tokių būdu galima paprastai įvesti tokias koordinates, kurios imituos žinomo gradiento paviršių. Pavyzdžiui, jei įvestume tokias koordinates: $z_{\mathrm{LF}} = z_{\mathrm{LM}} = z_{\mathrm{LH}} = 0$ mm, $z_{\mathrm{RF}} = z_{\mathrm{RM}} = z_{\mathrm{RH}} = 50$ mm, tai akivaizdu, kad robotas stovi ant šlaito kuris kyla jam iš dešinės. Taip pat nesunkiai galima suskaičiuoti tokio paviršiaus gradientą, kuris šiuo atveju yra lygus 11,3º , ir palyginus su programos suskaičiuotu kampu, galima įsitikinti skaičiavimo metodo teisingumu.

Toliau įvedamos tokios pėdų koordinatės: $z_{\mathrm{RF}} = z_{\mathrm{RH}} = z_{\mathrm{LM}} = -10$ mm, $z_{\mathrm{LF}} = z_{\mathrm{LH}} = -50$ mm, $z_{\mathrm{RM}} = 50$ mm. 3.3 pav. pavaizduota roboto būsena esant nurodytoms koordinatėms. Kaip matyti, robotas stovi ant paviršiaus, kuris kyla robotui iš dešinės. Suskaičiavus plokštumų pokrypio kampus, gaunamos tokios vertės: $\alpha_1 = \beta_1 = \beta_2 = 0$, $\alpha_2 = 21{,}8°$.



a) b)

3.3 pav. Roboto būsena, kai pėdų koordinatės $z_{\mathrm{RF}} = z_{\mathrm{RH}} = z_{\mathrm{LM}} = -10$ mm, $z_{\mathrm{LF}} = z_{\mathrm{LH}} = -50$ mm, $z_{\mathrm{RM}} = 50$ mm, prieš kūno korekciją, a) ir po b).

Fig. 3.3. Robot state with feet coordinates $z_{\mathrm{RF}} = z_{\mathrm{RH}} = z_{\mathrm{LM}} = -10$ mm, $z_{\mathrm{LF}} = z_{\mathrm{LH}} = -50$ mm, $z_{\mathrm{RM}} = 50$ mm, before corection (a) and after corection (b)

Iš 3.3 paveikslo (a) matyti, kad kairė priekinė ir galinė kojos yra beveik maksimaliai ištiestos, t. y. pėdos *z* koordinatės ir kojos pagrindo *z* koordinatės skirtumas yra 155 mm ir yra artimas maksimaliam roboto praeinamumui. Tai reiškia, kad roboto koja turi tik 5 mm judesio laisvę *z* ašimi. Jei robotui nekeliama užduotis išlaikyti kūną horizontaliai, tai paprasčiausias būdas padidinti pėdos judesių laisvę *z* ašimi yra pasukti roboto kūną ta pačia kryptimi kaip ir paviršiaus gradientas. Kampą, kuriuo reiktų pakreipti kūną galima apskaičiuoti kaip abiejų plokštumų pokrypio kampų vidurkį (3.1), (3.2).

$$\alpha_0 = -(\alpha_1 + \alpha_2)/2 \tag{3.1}$$

$$\beta_0 = -(\beta_1 + \beta_2)/2\,. \tag{3.2}$$

Pakreipus kūną kampu α_0 minėtų kojų judesio laisvė padidėja iki 14 mm (2.3 pav. (b)).

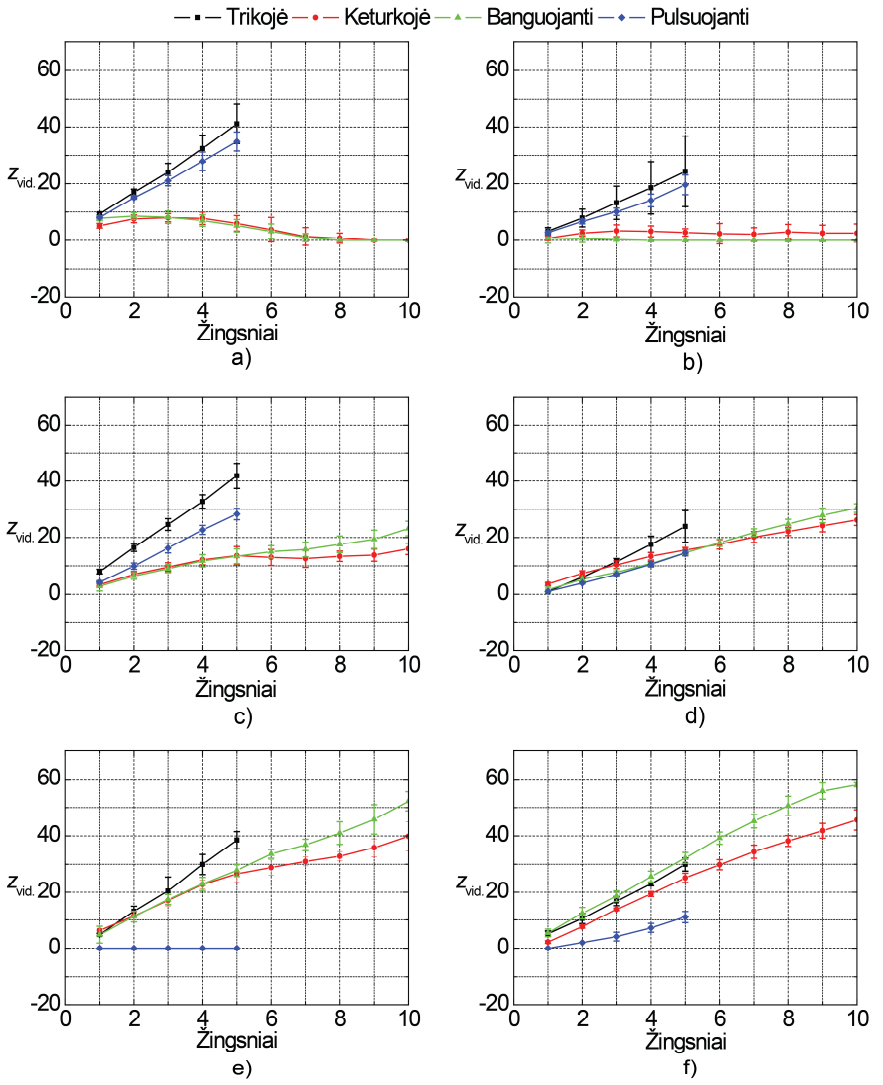
3.2. Šešiakojo roboto ėjimo tyrimas ir koordinačių nuokrypų kompensavimas

Robotui einant lygiu paviršiumi, prisitaikymas prie jo yra nesvarbus, nes žinoma, kad robotas kojas visada pastatys tokiame pačiame aukštyje, kitaip tariant pastatymo vieta yra visada žinoma. Tačiau, ėjimas nelygiu paviršiumi nulemia, kad kojos pastatymo vietos ir aukščiai nebus išlanksto žinomi. Todėl robotas turi gebėti prisitaikyti prie paviršiaus nelygumų, t. y. sustabdyti kojas, kai pėdos paliečia paviršių.

Tačiau, ėjimas prisitaikant prie paviršiaus nulemia, kad roboto pėdų aukščio koordinatės ne visada sutaps su paviršiaus nelygumų aukščiais. Šie nesutapimai atsiranda dėl roboto konstrukcijos mechaninių paklaidų (konstrukcinių dalių tamprumo, pavarų reduktorių laisvumo, galinių kontaktų eigos ir t. t.), pėdų praslydimo. Todėl netgi einant lygiu paviršiumi, taikant adaptyvų roboto judėjimą, atsiranda pėdų *z* koordinačių nuokrypos ir neteisingai interpretuojamas paviršiaus nelygumas. Norint užtikrinti teisingą paviršiaus atpažinimą, kuriuo juda robotas, reikia įvertinti ir kompensuoti minėtas nuokrypas.

Norint įvertinti nuokrypas, robotas turi eiti žinomu paviršiumi, kuris būtų atskaitos taškas pagal kurį bus vertinamas nuokrypų dydis. Todėl bandymai atliekami robotui einant visiškai lygiu paviršiumi, tokiu būdu yra žinoma, kad pėdos kiekvienu žingsniu bus pastatomos į neutralią padėtį ($z_i = 0$) ir bet koks nukrypimas nuo šios sąlygos bus nuokrypa. Kiekviena eisena bandymai atliekami

po 5 kartus, robotui paeinant 5 (trikoje ir pulsuojančia eisenomis) ir 10 (dvikoje ir banguojančia eisenomis) žingsnių.



3.4 pav. Pėdų *z* koordinačių kitimas, robotui einant skirtingomis eisenomis. a) kairė priekinė, b) dešinė priekinė, c) kairė vidurinė, d) dešinė vidurinė, e) kairė galinė, f) dešinė galinė kojos

Fig. 3.4. Feet *z* coordinate variation during robot locomotion using different gaits. a) left front, b) right front, c) left middle, d) right middle, e) left hind, f) right hind legs

Žingsnių skaičius skiriasi dėl to, kad dėl nuokrypų sumavimosi minėtomis eisenomis robotas daugiau žingsnių negali padaryti. Robotui einant po kiekvieno žingsnio pėdų koordinatės yra įrašomos į išorinę EEPROM atmintį.

3.4 paveiksle pateiktas visų pėdų *z* koordinačių vidurkio kitimas kas žingsnį kiekvienai eisenai. Galima pastebėti, kad daugumos kojų koordinatės didėja. Taip yra dėl to, kad kiekvieno žingsnio metu dėl minėtų nuorypų roboto korpusas truputi nusileidžia žemyn. Ir kadangi roboto korpuso koordinačių sistema yra susijusi su pėdų koordinačių sistema, kojos tariamai pastatomos aukščiau nei turėtų. Kadangi yra žinoma, kad pėdų koordinatės kiekvienu žingsnių turi būti lygios 0, tai nesunku apskaičiuoti kiek pasikeičia kiekvienos pėdos koordinatė. Šie pėdų koordinačių pokyčiai apskaičiuojami atimant esamą koordinatę iš prieš tai buvusios.

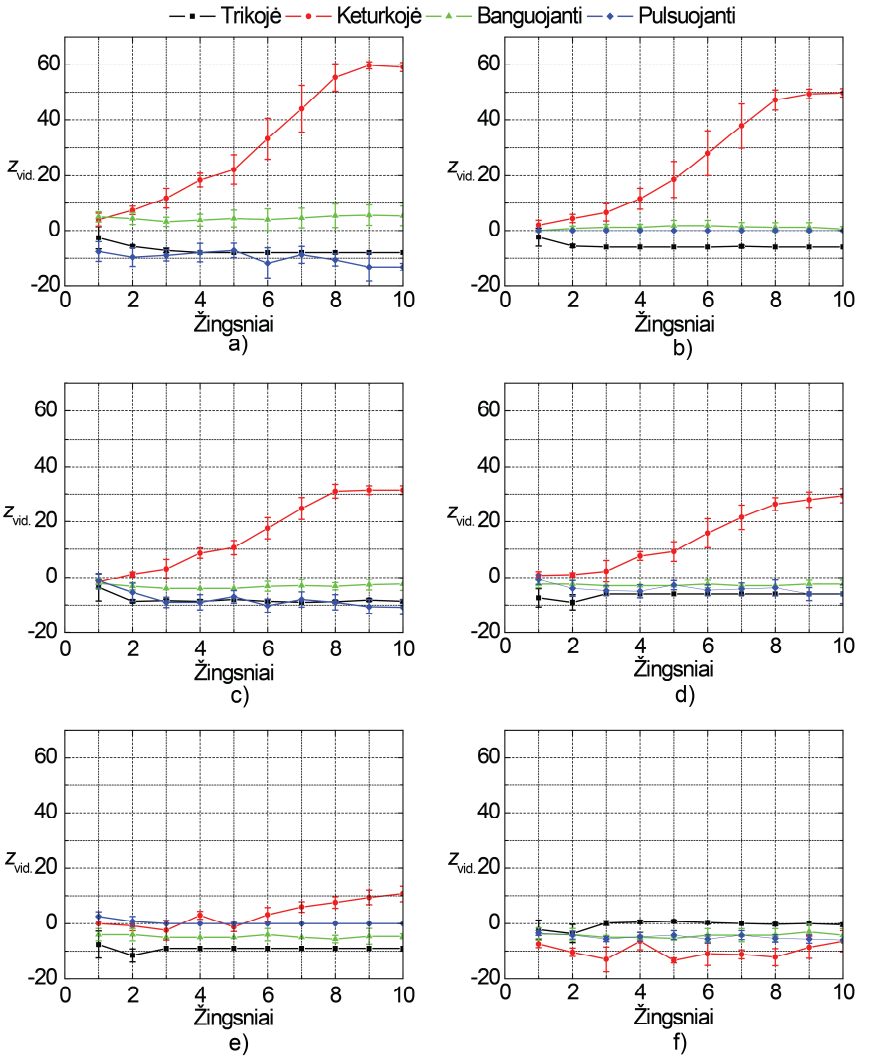
3.1 lentelė. Roboto pėdų *z* koordinačių skirtumų vidurkiai
Table 3.1. Robot‘s feet *z* coordinate difference means

	Pėdų *z* koordinačių skirtumai, mm					
Eisena	RF	RM	RH	LF	LM	LH
Trikojė	5,6±2,5	5,8±1,4	6,3±0,8	7,9±1,7	8,5±1,2	8,5±0,7
Keturkojė	1,1±0,3	2,4±0,2	4,7±0,2	1,3±0,4	2,1±0,3	3,9±0,5
Banguojanti	0,1±0,2	3,2±0,3	5,9±0,1	1,1±0,3	2,2±0,2	5,3±0,3
Pulsuojanti	4,2±0,9	3,4±0,2	2,8±0,5	6,7±0,8	6,1±0,5	0

3.1 lentelėje pateikiamos vidutinės vertės kiekvienai pėdai, parodančios kiek *z* koordinatės nukrypsta nuo reikiamos vertės (šiuo atveju 0). Šie duomenys parodo kiek reikia kiekvieną koją nuleist papildomai po kiekvieno žingsnio, norint išlaikyti stabilų ėjimą.

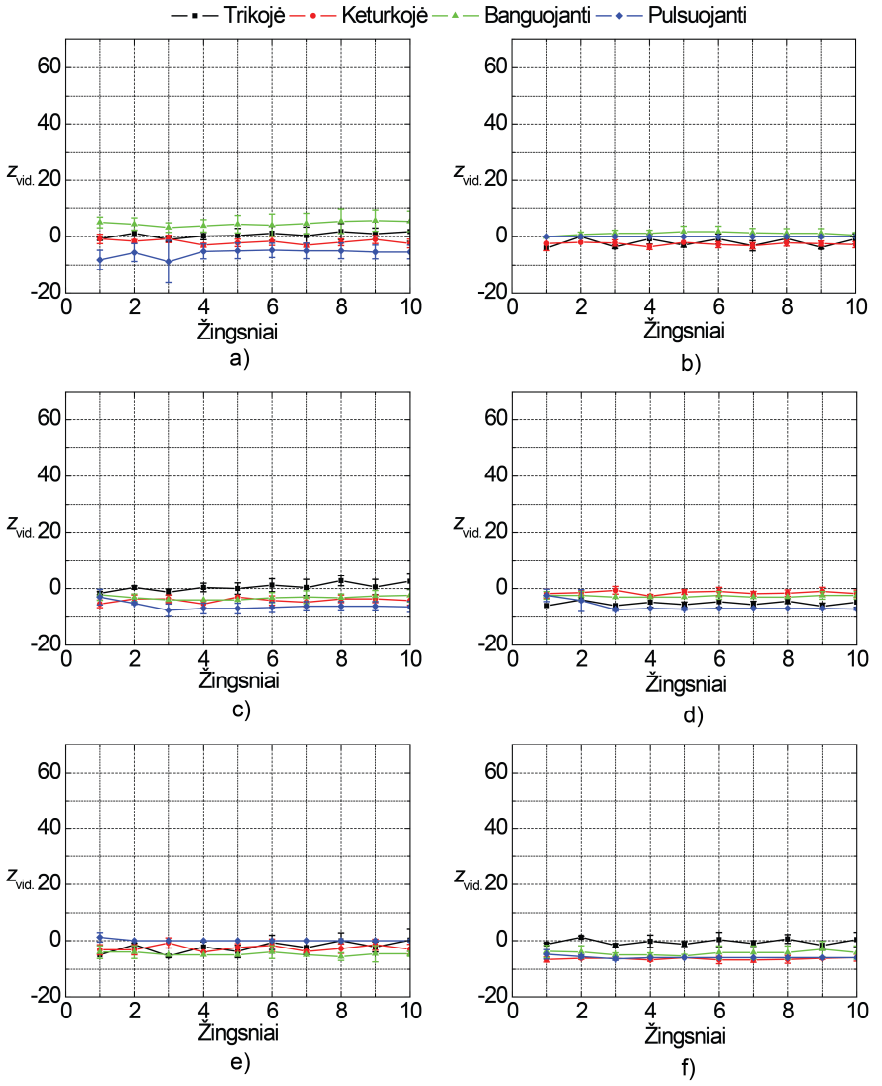
Įvedus į roboto valdymo sistemą papildomą šių nuokrypų kompensavimą, atliekami bandymai robotui einant 10 žingsnių visiškai lygiu paviršiumi, kiekvienas bandymas kartojamas 5 kartus.

3.5 paveiksle pateiktas pėdų *z* koordinačių kitimas robotui einant visiškai lygiu paviršiumi, kai roboto valdymo sistema kompensuoja nuokrypas. Matoma, kad trikojės, dvikojės ir banguojančios eisenų atvejais pėdų koordinatės nedidėja, išlieka tokios pat. Pagal 3.2 lentelės duomenis matyti, kad pėdų koordinačių skirtumų vidurkiai ženkliai sumažėja įvedus paklaidų kompensavimą. Taigi, roboto ėjimas tampa stabilesnis ir tokiu atveju roboto korpusas išlieka horizontalioje padėtyje ir nenusileidžia ant paviršiaus.



3.5 pav. Kojų *z* koordinačių kitimas, robotui einant skirtingomis eisenomis kompensuojant nuokrypas. a) kairė priekinė, b) dešinė priekinė, c) kairė vidurinė, d) dešinė vidurinė, e) kairė galinė, f) dešinė galinė kojos

Fig. 3.5. Feet *z* coordinate variation during robot locomotion using different gaits and compenstated deviations. a) left front, b) right front, c) left middle, d) right middle, e) left hind, f) right hind legs



3.6 pav. Kojų *z* koordinačių kitimas, robotui einant skirtingomis eisenomis kompensuojant nuokrypas, koeficientai pakoreguoti. a) kairė priekinė, b) dešinė priekinė, c) kairė vidurinė, d) dešinė vidurinė, e) kairė galinė, f) dešinė galinė kojos

Fig. 3.6. Feet *z* coordinate variation during robot locomotion using different gaits and corrected compenstated deviations. a) left front, b) right front, c) left middle, d) right middle, e) left hind, f) right hind legs

3.2 lentelė. Roboto pėdų *z* koordinačių skirtumų vidurkiai po kompensavimo
Table 3.2. Robot's feet *z* coordinate means after compensation

Eisena	Pėdų *z* koordinačių skirtumai, mm					
	RF	RM	RH	LF	LM	LH
Trikojė	0,5±0,3	1±0,4	1,4±0,4	0,6±0,3	1,1±0,5	1,1±0,6
Keturkojė	5,2±0,9	3,3±0,3	4,7±0,3	6,1±0,4	3,8±0,2	3,2±0,4
Banguojanti	0,4±0,3	0,4±0,6	0,7±0,7	1,2±0,5	0,5±0,7	0,7±1,1
Pulsuojanti	0	1,7±0,4	1,9±0,5	4,5±0,9	3,5±0,7	0,4±0,2

Einant pulsuojančia eisena pėdų koordinatės vis dar kinta ir neišlaikoma pastovi *z* koordinatė. Visi atsirandantys koordinačių nukrypimai nuo 0 gali būti aiškinami taip, kad kompensuojant paklaidas, dėl kompensavimo koeficientų apvalinimo atsiranda netikslumai. Taip pat dėl šios priežasties pėdų *z* koordinatės tampa perkompensuotos ir apytiksliai lygios nuo 0 iki –10 mm. Todėl kompensavimo koeficientus reikia paderinti eksperimentiškai, kad roboto ėjimas būtų stabilesnis.

3.3 lentelė. Roboto pėdų *z* koordinačių skirtumų vidurkiai po pakoreguoto kompensavimo
Table 3.3. Robot's feet *z* coordinate means after corrcted compensation

Eisena	Pėdų *z* koordinačių skirtumai, mm					
	RF	RM	RH	LF	LM	LH
Trikojė	3,4±0,5	2±0,2	2,4±0,2	1,9±0,3	2,2±0,4	3,3±0,5
Keturkojė	1,5±0,4	1,8±0,2	1±0,3	1,6±0,4	2,1±0,4	2,6±0,1
Banguojanti	0,4±0,3	0,4±0,6	0,7±0,7	1,2±0,5	0,5±0,7	0,7±1,1
Pulsuojanti	0	0,6±0,3	0,2±0,3	1,6±0,87	0,8±0,8	0,1±0,2

Eksperimentiškai paderinus kompensavimo koeficientus, pėdų koordinačių vidutinės vertės dar labiau sumažėja ir priartėja prie 0 (3.6 pav.). Pakoreguoti buvo tik trikojės, dvikojės ir pulsuojančios eisenų koeficientai, nes banguojančios eisenos pėdų koordinatės kito mažai ir papildoma kompensavimo koeficientų korekcija nebuvo reikalinga. Lentelėje 3.3 pateikiamos galutinės pakoreguotos kompensavimo koeficientų vertės. Lentelėje pateikiamos banguojančios eisenos pėdų koordinačių skirtumai yra paimti iš 3.2 lentelės. Naudojant 3.3 lentelės kompensavimo koeficientus roboto valdymo sistemoje, gaunamas stabilus roboto ėjimas lygiu paviršiumi taikant adaptyvų ėjimo režimą.

3.3. Eksperimentinis paviršiaus nelygumo vertinimo tyrimas

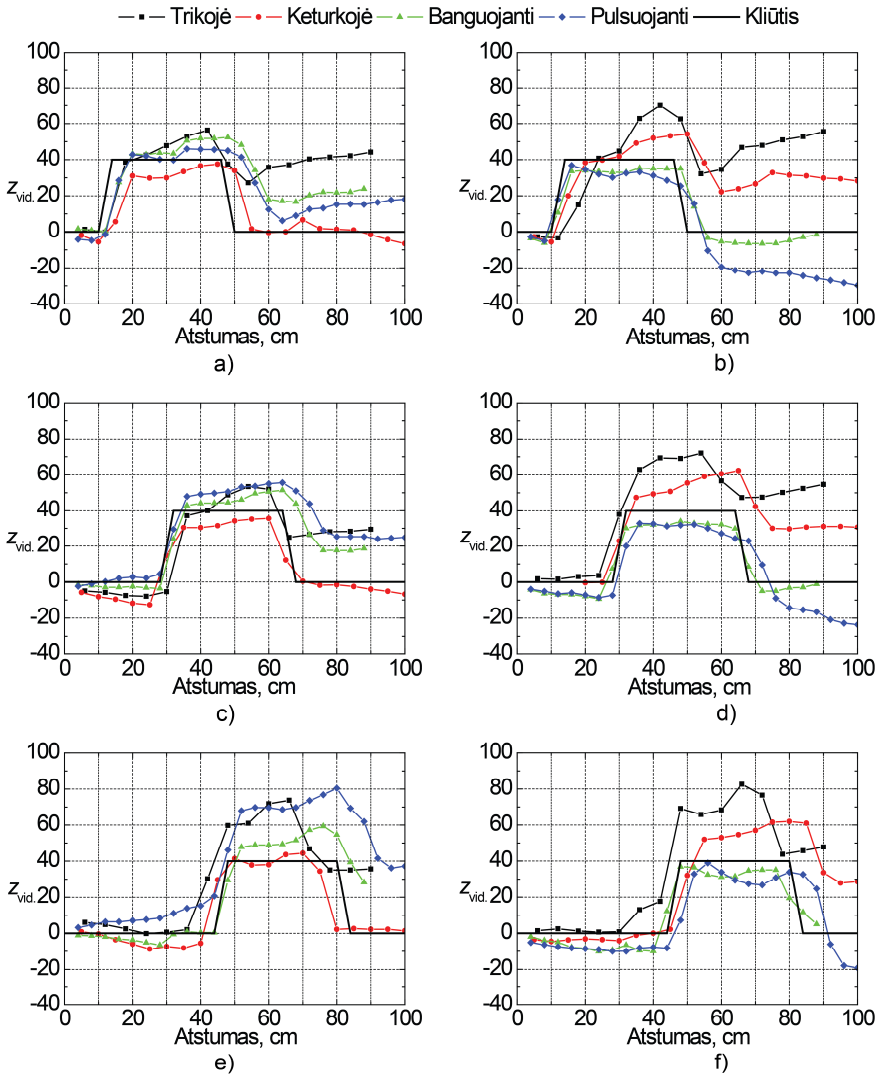
Paviršiaus nelygumo vertinimo metodo patikrinimui atliekami bandymai robotui einant per žinomų parametrų (aukščio ir ilgio) kliūtį. Kliūtis yra stačiakampė (40 mm aukščio ir 400 mm ilgio). Atliekant bandymus su žinomų parametrų kliūtimi yra nesudėtinga įvertinti metodo teisingumą. Kiekvienas bandymas atliekamas po penkis kartus su kiekviena eisena. Kiekvienos pėdos aukščio koordinatės yra įrašomos į išorinę EEPROM atmintį.

3.7 paveiksle parodytos visų pėdų *z* koordinačių vidutinės vertės robotui einant per minėtą stačiakampę kliūti skirtingomis eisenomis naudojant pastovius pėdų koordinačių nuokrypų kompensavimo koeficientus. Galima aiškiai pastebėti, kad robotui pradedant lipti ant kliūties koordinatės pradeda didėti ir apytiksliai užkyla iki 40 mm. Robotui nuėjus apytiksliai 40 cm pastebimas koordinačių mažėjimas. Tačiau pastebimi itin dideli nukrypimai nuo tikrųjų verčių. Iš grafikų matoma, kad didžiausia koordinačių nuokrypa yra 43±9 mm. Šios nuokrypos nuo tikrųjų verčių taip pat sietinos su minėtomis paklaidomis.

Nevienodos skirtingų pėdų *z* koordinačių nuokrypos rodo, kad roboto kūnas neišlieka horizontalus. Iš 3.7 paveikslo matyti, kad kairės priekinės ir kairės vidurinės kojų *z* koordinatės (3.7 pav., a ir c) nukrypsta mažiau nei kitų pėdų koordinatės (3.7 pav., b, d–f). Iš to galima teigti, kad roboto kūnas pakrypsta atgal ir į dešinę.

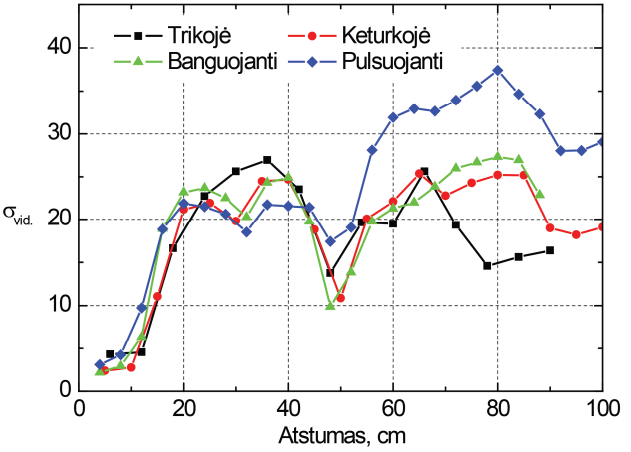
Einant per stačiakampę kliūti σ turi būti didesnis už 0 tik tol, kol robotas lipa ant arba nuo kliūties. Kai visos roboto kojos atsiduria ant kliūties, σ turi būti lygus 0. Iš 3.8 paveikslo matyti, kad kol robotas pradeda lipti ant kliūties σ yra artimas 0, tačiau robotui pradėjus lipti ant kliūties ima didėti, kadangi roboto pėdų koordinatės nebėra vienoje plokštumoje ir vis labiau pradeda išsibarstyti. Artėjant tam momentui, kai visos roboto kojos užkeliamos ant kliūties σ vertė pradeda mažėti, tačiau nesumažėja iki 0, nes kliūties plotis nėra ženkliai didesnis už roboto ilgį ir ne visos pėdos vienu metu būna pastatytos ant kliūties. Robotui pradėjus nulipinėti nuo kliūties vėl pastebimas σ vertės didėjimas, nes pėdų *z* koordinatės vis labiau išsibarsto. Didžiausia σ nuokrypa yra 37±3 mm (3.8 pav. ties 80 cm), maksimali vertė gali būti lygi 20 mm, kai trys roboto pėdos yra ant kliūties, 40 mm aukštyje, o kitos ant žemės, 0 mm.

Kadangi paklaidų kompensavimo koeficientai yra suderinti ėjimui lygiu paviršiumi, robotas negali išlaikyti stabilaus ėjimo esant nelygiam paviršiui. Tai parodo, kad kompensavimo koeficientai neturėtų būti pastovūs, o priklausyti nuo aplinkos sąlygų ir kisti keičiantis paviršiaus nelygumui.



3.7 pav. Pėdų *z* koordinačių vidutinės vertės einant per stačiakampę kliūti skirtingomis eisenomis, neapibrėžtis ±9 mm. a) kairė priekinė, b) dešinė priekinė, c) kairė vidurinė, d) dešinė vidurinė, e) kairė galinė, f) dešinė galinė kojos

Fig. 3.7. Feet *z* coordinate average values during locomotion over rectangular obstacle using different gaits, uncertainty ±9 mm. a) left front, b) right front, c) left middle, d) right middle, e) left hind, f) right hind legs



3.8 pav. Standartinės nuokrypos kitimas robotui einant per stačiakampę kliūti skirtingomis eisenomis, neapibrėžtis ±3 mm
Fig. 3.8. Standard deviation variation during robot locomotion over rectangular obstacle using different gaits, uncertainty ±3 mm

Iš 3.7 paveikslo matyti, kad pėdų *z* koordinatės nukrypsta nuo kliūties formų daugiausiai einant trikoje ir keturkoje eisenomis, mažiau nukrypsta einant banguojančia ir pulsuojančia eisenomis. Tai parodo, kad robotas yra stabilesnis (kūnos horizontalesnis) einant lėtesnėmis eisenomis. Tai taip pat pagrindžia eisenų reitingų priskirimą, vertinant jas pagal stabilumą sprendimų priėmimo matricoje.

Galima pastebėti roboto kūno pokrypio ryšį su pėdų *z* koordinačių nuokrypomis, todėl pėdų nuokrypas galima kompensuoti pagal roboto kūno pokrypį. Šiam tikslui reikia žinoti roboto pokrypį laipsniais *x* ir *y* ašimis, kuris išmatuojamas akcelerometru. Pėdų *z* koordinačių nuokrypas pagal roboto kūno pokrypius galima apskaičiuori pagal (3.3)–(3.8) formules.

$$z_{e,\mathrm{RF}} = \overline{z} + \frac{L1 \cdot \sin\left(\alpha_x\right)}{2} - \frac{L2 \cdot \cos\left(\alpha_x\right) \cdot \sin\left(\alpha_y\right)}{2}, \tag{3.3}$$

$$z_{e,\mathrm{RM}} = \overline{z} - \frac{L2 \cdot \cos\left(\alpha_x\right) \cdot \sin\left(\alpha_y\right)}{2}, \tag{3.4}$$

$$z_{e,\mathrm{RM}} = \overline{z} - \frac{L1 \cdot \sin(\alpha_x)}{2} - \frac{L2 \cdot \cos(\alpha_x) \cdot \sin(\alpha_y)}{2}, \tag{3.5}$$

$$z_{e,\mathrm{LF}} = \overline{z} + \frac{L1 \cdot \sin(\alpha_x)}{2} + \frac{L2 \cdot \cos(\alpha_x) \cdot \sin(\alpha_y)}{2}, \tag{3.6}$$

$$z_{e,\mathrm{LM}} = \overline{z} + \frac{L2 \cdot \cos(\alpha_x) \cdot \sin(\alpha_y)}{2}, \tag{3.7}$$

$$z_{e,\mathrm{LH}} = \overline{z} - \frac{L1 \cdot \sin(\alpha_x)}{2} + \frac{L2 \cdot \cos(\alpha_x) \cdot \sin(\alpha_y)}{2}, \tag{3.8}$$

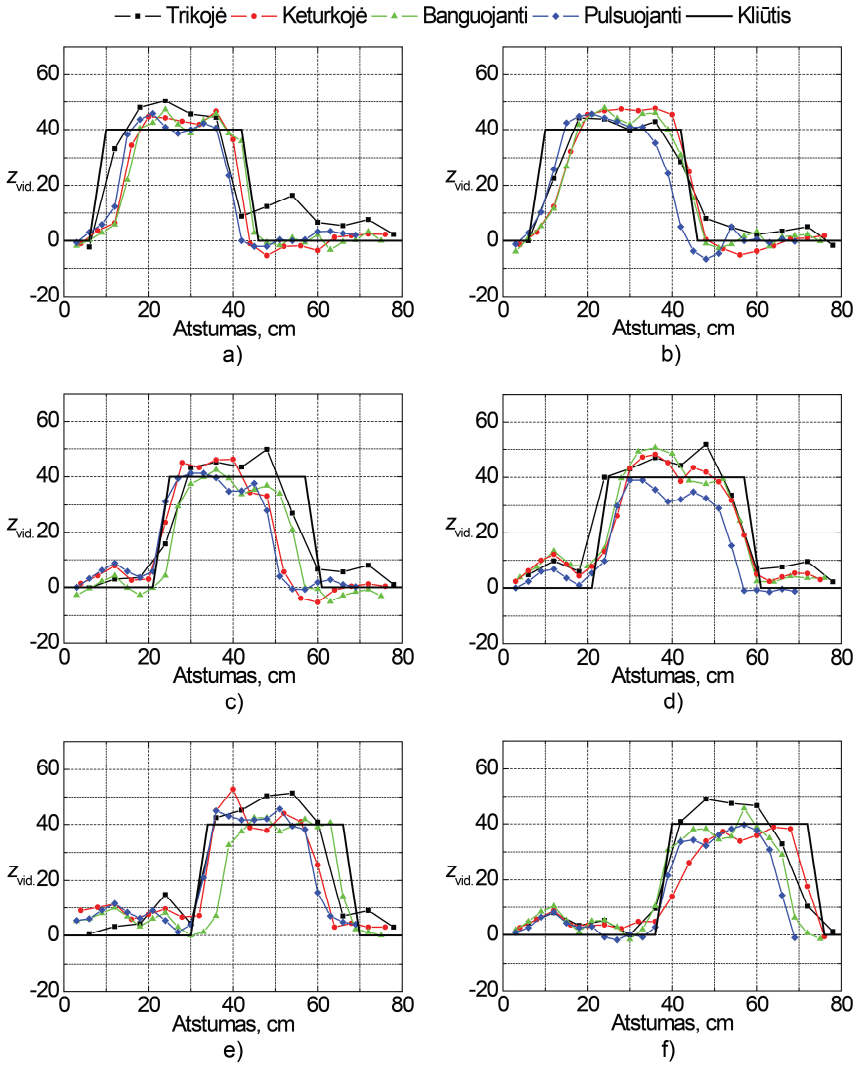
čia $\overline{z}$ – pėdų *z* koordinačių vidurkis, perskaičiuojamas po kiekvieno žingsnio, α_x, α_y – roboto kūno pokrypiai *x* ir *y* ašimis, išmatuojami akcelerometru.

Po kiekvieno roboto žingsnio akcelerometru išmatuojami roboto kūno pokrypiai *x* ir *y* ašimis ir apskaičiuojamas pėdų *z* koordinačių vidurkis. Ir pagal (3.3)–(3.8) formules randami kompensavimo koeficientai.

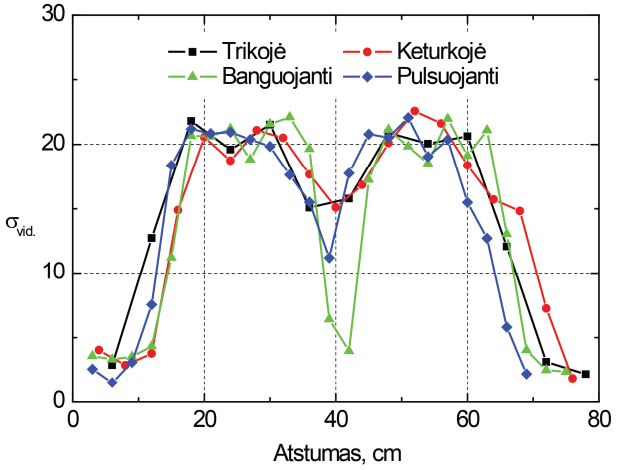
Taikant tokią pėdų koordinačių kompensaciją atlikti 5 bandymai su kiekviena eisena robotui einant per stačiakampę kliūtį. Iš rezultatų (3.9 pav.) matyti, kad pėdų koordinatės pradeda didėti robotui lipant ant kliūties ir mažėja nulipant nuo kliūties. Išmatuotos pėdų *z* koordinatės yra artimos realiai kliūties formai bei matmenims. Iš grafikų matyti, kad visų pėdų *z* koordinatės kinta vienodai, t. y. nei vienos pėdos koordinatės išskirtinai daug nenukrypsta nuo tikrųjų verčių. Todėl roboto kūnas išlieka horizontalus ir robotas juda stabiliai, geba įveikti paviršiaus nelygumus jų neužkliudydamas. Didžiausia pėdų koordinatės nuokrypa siekia 16±4 mm kas parodo paviršiaus nelygumų aukščio atpažinimo tikslumą.

Lyginant su nuokrypų kompensavimu naudojant pastovius pėdų *z* koordinačių kompensavimo koeficientus, paviršiaus nelygumų aukščio atpažinimas, taikant kompensavimą pagal roboto kūno pokrypius, pagerėja iki 2,7 karto, o neapibrėžtys sumažėja iki 2,2 karto.

Standartinė nuokrypa σ (3.10 pav.) didėja robotui lipant ant ir nuo kliūties, sumažėja, kai robotas yra užlipęs ant kliūties ir yra nulipęs nuo jos. Iš grafiko matyti, kad σ vertės prieš kliūti ir po jos yra panašios ir artimos 0, todėl galima teigti, kad σ kito teisingai ir iš jos vertės galima spręsti apie paviršiaus nelygumą.



3.9 pav. Pėdų z koordinačių vidutinės vertės einant per stačiakampę kliūti skirtingomis eisenomis nuokrypas koreguojant pagal roboto kūno pokrypius, neapibrėžtis ±4 mm. a) kairė priekinė, b) dešinė priekinė, c) kairė vidurinė, d) dešinė vidurinė, e) kairė galinė, f) dešinė galinė kojos,
Fig. 3.9. Feet *z* coordinate average values during locomotion over rectangular obstacle using different gaits with deviation compensation using robot body inclinations, uncertainty ±4 mm. a) left front, b) right front, c) left middle, d) right middle, e) left hind, f) right hind legs



3.10 pav. Standartinės nuokrypos kitimas robotui einant per stačiakampę kliūti skirtingomis eisenomis nuokrypas kompensuojant pagal roboto kūno pokrypius, neapibrėžtis ±2 mm
Fig. 3.10. Standard deviation variation during robot locomotion over rectangular obstacle using different gaits with deviation compensation using robot body inclinations, uncertainty ±2mm

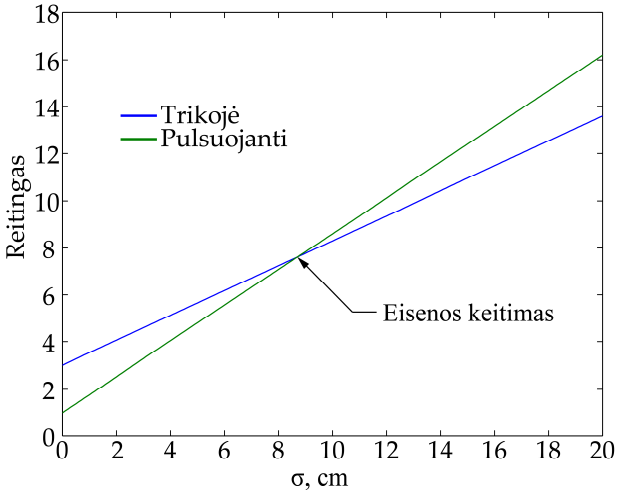
Standartinės nuokrypos vertė nesumažėja iki 0 ties atstumu 40 cm, kai robotas turi būti užlipęs ant kliūties (3.10 pav.) todėl, kad kliūties ilgis yra labai artimas roboto ilgiui ir labai maža tikimybė, kad visos roboto pėdos atsiduras ant kliūties tuo pat metu. Taip pat σ vertės maksimalios nuokrypos sumažėja iki 9,25 karto (iki 4 mm), o neapibrėžtys iki 1,5 karto (iki ±2 mm).

3.4. Eisenos parinkimo imitavimo rezultatai

Nagrinėjami tik kraštutiniai atvejai, trikojė ir pulsuojanti eisenos. Trijokė eisena yra laikoma greičiausia, bet mažiausiai stabilia ar atsargia, pulsuojanti eisena yra laikoma lėčiausia, tačiau atsargiausia ir stabiliausia.

3.11 paveikslas rodo, kad pulsuojančios eisenos reitingas didėja greičiau nei trikojės eisenos didėjant σ vertei. Greičio prioritetas nulemia, kad kol σ vertė yra maža, parenkama greitesnė eisena (šiuo atveju trikojė). Tačiau ties tam tikra σ verte pulsuojančios eisenos reitingas tampa didesnis už trikojės eisenos reitingą ir eisena yra keičiama į atsargesnę. Norint, kad greitesnė eisena būtų išlai-

koma ant didesnio nelygumo paviršių, eisenos keitimo taškas būtų ties didesne σ verte, užtenka padidinti trikojės eisenos prioritetą. σ verčių ribos tampa visiškai nesvarbios, nes kriterijų prioritetai nulemia eisenos keitimo taškus.

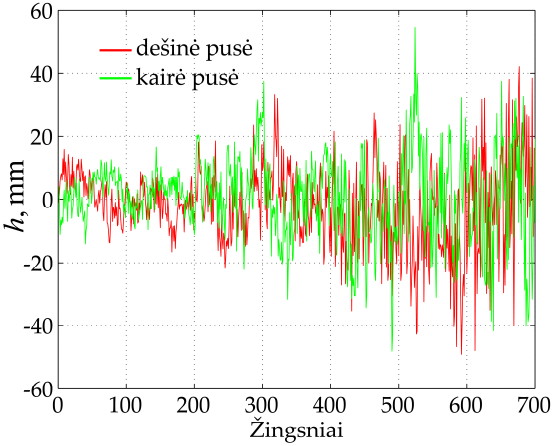


3.11 pav. Trikojės ir pulsuojančios eisenų reitingų kitimas, standartinė nuokrypa kinta tiesiškai. Linijų susikirtimo taškas parodo eisenos keitimo tašką. Kitas eisenos keitimo taškas gali būti gaunas keičiant eisenų prioritetus

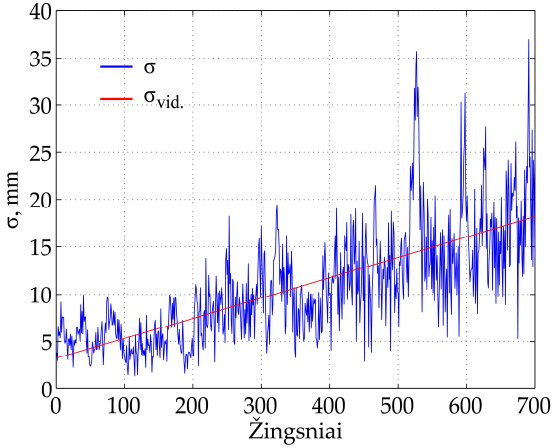
Fig. 3.11. Tripod and ripple gait rating variation, Standard deviation changes lineary. Line intersection point indicates gait change point. Different gait change point could be achieved by changing gait priorities

Roboto valdymo sistemos tikslas parinkti tinkamą eiseną priklausomai nuo užduoties reikalavimų ir atsižvelgiant į paviršiaus nelygumą. Valdymo sistema parenka naują eiseną, kai vienos iš eisenų reitingas tampa didesnis už kitų eisenų reitingus. Eisenos parinkimo patikrinimui sugeneruojamas paviršius su kintančiu paviršiaus nelygumu (3.12 pav.). Visų imitacijų metu, žingsnio ilgis yra 5 cm.

Kaip matyti iš 3.13 paveikslo, σ vertė didėja, tačiau vietinis paviršiaus nelygumas gana ženkliai kinta. Dėl šios priežasties eisenos parinkimas po kiekvieno žingsnio yra beveik neįmanomas, kadangi eisena bus keičiama labai dažnai. 3.14 paveiksle (a) pavaizduoti imitacijos rezultatai, kai eisena yra keičiama po kiekvieno žingsnio, kai pasikeičia eisenos reitingas. Yra akivaizdu, kad eisena keičiama labai dažnai, todėl robotas neišlaiko pastovaus, sklandaus ėjimo, bet nuolat keičia eiseną.



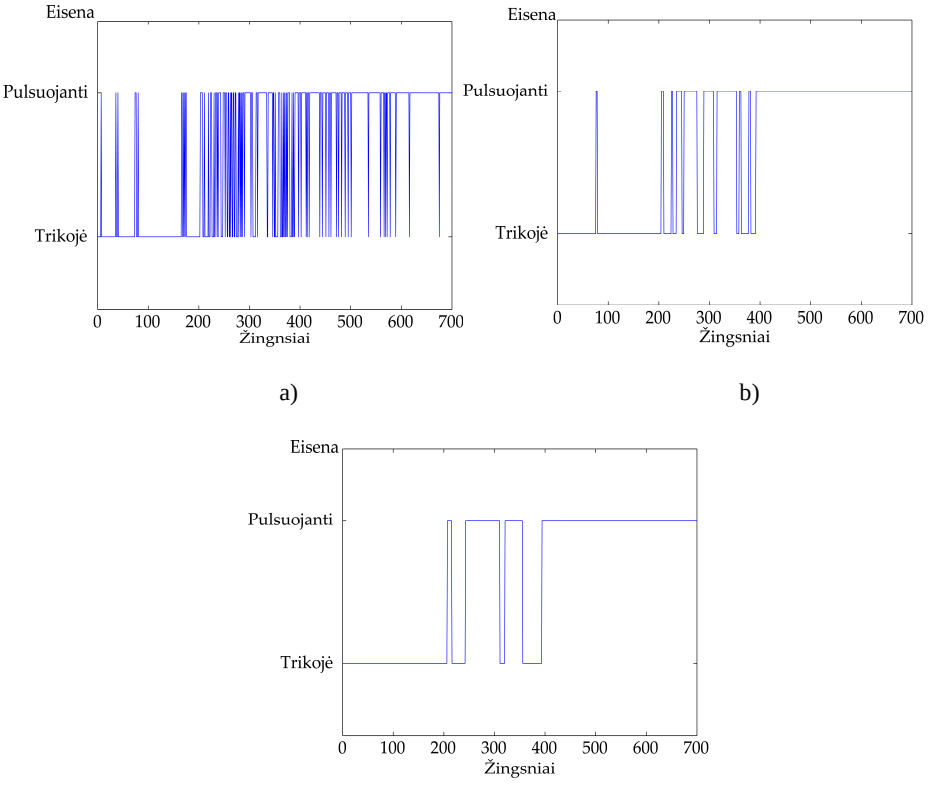
3.12 pav. Paviršiaus profilis kairėms ir dešinėms roboto kojoms
Fig. 3.12. Terrain profile for left and right robot legs



3.13 pav. Paviršiaus (3.12 pav.) standartinės nuokrypos profilis
Fig. 3.13. Terrain (Fig. 3.12) standard deviation profile

Viena iš galimybių, norint užtikrinti sklandesnį roboto ėjimą, yra keisti eiseną, jei eisenos reitingas (paviršiaus nelygumas) išlieka nepakitęs tam tikrą skaičių žingsnių. 3.14 paveiksle (b) parodyti eisenos parinkimo imitavimo rezultatai, kai eisena keičiama, jei eisenos reitingas išlieka nepakitęs tris žingsnius iš eilės. Galima pastebėti, kad eisenos keitimo dažnis sumažėjo. Tačiau, vis vien išlieka vietų kur naujai parinka eisena būna parinka labai trumpą laiką. Reiktų

atkreipti dėmesį į eisenos keitimo dažnio padidėjimą tarp 200 ir 400 žingsnių. Šioje vietoje eisenų reitingai tampa labai artimi, kadangi paviršiaus nelygumas yra ties eisenos keitimo tašku.



3.14 pav. Eisenos keitimas esant didėjančiam paviršiaus nelygumui. a) eisenos keitimas po kiekvieno žingsnio, b) eisenos keitimas po trijų žingsnių, c) eisenos keitimas po penkių žingsnių

Fig. 3.14. Gait change in increasing roughness terrain. a) gait change after each step, b) gait change after each three steps, c) gait change after each five steps

Eisenos keitimo dažnis dar labiau mažėja, jei eisena keičiama po penkių žingsnių (3.14 pav., c). Padidėjęs eisenos keitimo dažnis vis dar gali būti pastebimas tarp 200 ir 400 žingsnių.

Nors eisenos keitimo dažnis mažėja, ją keičiant po kelių žingsnių, tačiau tai reiškia, kad robotas, šiuos kelis žingsnius, turės eiti su netinkama eisena. Galima

daryti išvadą, kad didinant žingsnių skaičių eisenos pakeitimui, eisenos keitimo dažnis sumažėja ir eisena bus keičiama tik tada, kai paviršiaus nelygumas bus tikrai pasikeitęs. Tai taip pat leidžia atmesti staigius paviršaus nelygumų pakitimus (vienetiniai iškilimai, įdubimai ar netgi sistemos klaidos). Neigiama tokio eisenos parinkimo pusė yra ta, kad robotas šiuos kelis žingsnius paviršiumi eis neteisinga eisena.

3.5. Trečiojo skyriaus išvados

Atlikus šešiakojo žingsniuojančio roboto imitacinius tyrimus pastebėta, kad standartinės nuokrypos parametras gali būti taikomas paviršiaus nelygumų atpažinimui ir įvertinimui. Atliekant eksperimentinius roboto judėjimo tyrimus pastebėta, kad dėl roboto konstrukcijos atsiranda paklaidos vertinant pėdų koordinates, kurios įtakoja standartinės nuorypos vertę ir paviršiaus įvertinimo tikslumą.

1. Roboto pėdų *z* koordinačių standartinės nuokrypos vertės labiausiai priklauso tik nuo roboto pėdų aukščio koordinačių išsibarstymo, paviršiaus šlaito įtaka standartinei nuorypai yra nykstamai maža lyginant su paviršiaus nelygumu.
2. Taikant adaptyvią eiseną, dėl konstrukcijos netikslumo, reduktorių laisvumo ir kitų priežasčių, atsiranda pėdų *z* koordinačių nuokrypos. Dėl šios priežasties robotas eidamas lygiu paviršiumi praranda stabilumą (nukrypsta kūno padėtis). Pėdų koordinačių nuokrypos kinta nuo 5,6±2,5 mm iki 8,5±1,2 mm trikojei eisenai, nuo 1,1±0,3 mm iki 4,7±0,2 mm keturkojai eisenai, nuo 0,1±0,2 mm iki 5,9±0,1 mm banguojančiai eisenai, nuo 0 mm iki 6,7±0,8 mm pulsuojančiai eisenai.
3. Mechanines paklaidas galima kompensuoti valdymo sistemoje. Jos kompensuojamos kiekvieną roboto koją, pastačius ant paviršiaus, papildomai nuleidžiant žemyn. Kompensavus pėdų *z* koordinačių nuokrypos vidutiniškai sumažėja 65 % trikojei, 31 % keturkojei, 77 % banguojančiai ir 85 % pulsuojančiai eisenoms.
4. Eksperimentiškai ištyrus roboto ėjimą per stačiakampę kliūtį, taikant pastovius nuokrypų kompensavimo koeficientus, pastebėta, kad pėdų *z* koordinatės didėja robotui lipant ant kliūties ir mažėja nulipant. Tačiau dėl roboto mechaninių paklaidų, pėdų praslydimų atsiranda koordinačių nuokrypos ir maksimali nuokrypa yra 43±9 mm. Todėl pėdų *z* koordinačių standartinė nuokrypa taip pat netiksliai įvertina paviršiaus nelygumus ir maksimali nuokrypa yra 37±3 mm.

5. Eksperimentiškai nustatyta, kad taikant nuokrypų kompensavimą pagal roboto kūno pokrypius paviršiaus nelygumų aukščio atpažinimo tikslumas pagerėja iki 2,7 karto, neapibrėžtys sumažėja iki 2,2 karto, o standartinės nuokrypos neatitikimas sumažėja iki 9,25 karto, o neapibrėžtys – iki 1,5 karto.
6. Pasiūlytas eisenos parinkimo algoritmas leidžia parinkti roboto eisenų priklausomai nuo pėdų *z* koordinačių standartinės nuokrypos vertės bei kitų roboto užduočiai keliamų reikalavimų. Siūlomas eisenos parinkimo metodas leidžia atsiriboti nuo paviršiaus nelygumo skirstymo į kategorijas, tačiau eisena turi būti parenkama jei standartinė nuokrypa nesikeičia mažiausiai penkis žingsnius.

Bendrosios išvados

1. Šešiakojų žingsniuojančių robotų kinematiką patogu spręsti taikant homogenines transformacijų matricas bei atvirkštinę kinematiką vienai kojai. Sudarius pėdų trajektorijų generavimo matematines išraiškas ir apjungus jas su roboto kinematiniu modeliu galima sudaryti supaprastintą roboto valdymo algoritmą, užtikrinantį labai paprastą šešiakojo roboto valdymą, leidžiantį roboto judesius valdyti realiu laiku, nepaveikiant jo ėjimo.
2. Pateikiama paviršiaus nelygumų vertinimo metodika taikant standartinės nuokrypos parametrą, kuris suskaičiuojamas pagal esamas roboto pėdų aukščio koordinates, leidžia labai paprastai įvertinti paviršiaus nelygumus. Tačiau yra jautrus kiekvienos kojos koordinačių nuokrypoms, kurios gali iškreipti paviršiaus įvertinimą.
3. Pasiūlytas naujas paviršiaus nelygumo ir pokrypio įvertinimo metodas skaičiuojant kampus tarp dviejų plokštumų nubrėžtų per roboto pėdas. Apskaičiavus kampus tarp vertikalės, nustatytos pagal akcelerometro duomenis, ir šių plokštumų, kampų skirtumas įvardina paviršiau nelygumą, o vidurkis – pokrypį.
4. Pėdų koordinačių nuokrypas galima sumažinti kompensuojant esamas mechanines paklaidas valdymo sistemoje nurodant papildomai

nuleisti kiekvieną koją žemiau. Tai leidžia pėdų koordinačių nuokrypas, einant lygiu paviršiumi, sumažinti 65 % trikojei, 31 % keturkojei, 77 % banguojančiai ir 85 % pulsuojančiai eisenoms.

5. Eksperimentiškai nustatyta, kad kompensuojant pėdų *z* koordinačių nuokrypas taikant pastovius kompensavimo koeficientus, paviršiaus nelygumų aukštis įvertinamas su 43±9 mm nuokrypa, o σ verčių nuokrypos siekia 37±3 mm. Nuokrypas kompensuojant pagal roboto kūno pokrypius, išmatuojamus akcelerometru, nelygumų aukščio atpažinimas pagerėja iki 2,7 karto, neapibrėžtys sumažėja iki 2,2 karto, standartinės nuokrypos neatitikimas sumažėja iki 9,25 karto, neapibrėžtys – iki 1,5 karto.
6. Sudaryta roboto eisenos parinkimo metodas leidžia neapsiriboti iš anksto apibrėžtomis paviršiaus nelygumo kategorijomis. Šis metodas taip pat užtikrina, kad roboto eisenos keitimas priklausys ir nuo roboto užduočiai keliamų reikalavimų, tokių kaip greitis, stabilumas ar atsargumas. Kaip pagrindinis šio metodo trūkumas gali būti įvardijamas eisenos keitimas po kelių žingsnių, kas užtikrina, kad paviršiaus nelygumas tikrai tapo didesnis ar mažesnis, nei anksčiau. Bet tuo pačiu ir užtikrina, kad roboto sprendimų priėmimo sistema nereaguos į vienetinius, didesnius nelygumo pokyčius, tai sumažina standartinės nuokrypos jautrumą į vienetinius didelius pėdų koordinačių pokyčius.

Literatūra ir šaltiniai

Bajd, T.; Mihelj, M.; Lenarcic, A.; Stanovnik, A.; Munih, M. 2010. *Robotics (Intelligent Systems, Control and Automation: Science and Engineering)*. Springer: Netherlands. ISBN 978-90-481-3776-3. 152 p.

Baškys, B.; Fedaravičius, A. 2004. *Robotų technika*. Kaunas: Technologija. ISBN 9955-09-802-3. 494 p.

Beauchemin, S. S.; Barron, J. L. 1995. The computation of optical flow. *ACM Computing Surveys* 27(3): 433–466.

Berns, K.; Ilg, W.; Deck, M.; Albiez, J.; Dillman, R. 1999. Mechanical construction and computer architecture of the four-legged machine BISAM. *IEEE Trans. on Mechatronics* 4(1), 1–7.

Botello, O. E. L.; García, M. L. B.; del Carmen Zambrano Robledo, P.; Velázquez, A. R. R. 2010. Design of a Hexapod Robot Based on Insects. *Electronics, Robotics and Automotive Mechanics Conference (CERMA)*. 347–354.

Browning, B.; Deschaud, J. E.; Prasser, D.; Rander, P. 2012. 3D Mapping for high-fidelity unmanned ground vehicle lidar simulation. *The International Journal of Robotics Research* 31(12): 1349–1376.

Campbell, B. A.; Garvin, J. B. 1993. Lava flow topographic measurments for radar data interpretation. *Geophysical Research Letters* 20(9): 831–834.

Carpin, S.; Lewis, M.; Wang, J.; Balakirsky, S.; Scrapper, C. 2007. USARSim: a robot simulator for research and education. In: IEEE International Conference on Robotics and Automation: 1400–1405.

Castelnovi, M.; Arkin, R.C.; Collins, T.R. 2005. Reactive Speed Control System Based on Terrain Roughness Detection. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*: 891–896.

Celaya, E.; Luis Albarral, J. 2003. Implementation of a Hierarchical Walk Controller for the LAURON III Hexapod Robot. *Proceedings of 6th International Conference on Climbing and Walking Robots (CLAWAR 2003)*. 409–416.

Chow, Y. H.; Chung, R. 2002. VisionBug: A Hexapod Robot Controlled by Stereo Cameras. *Autonomous Robots* 13(3): 259–276.

Davis, C. J. ; Reisinger, T. W. 1990. Evaluating Terrain for Harvesting Equipment Selection. *International Journal of Forest Engineering* 2(1): 9–16.

Erden, M. S.; Leblebicioğlu, K. 2005. Multi legged walking in robotics and dynamic gait pattern generation for a six-legged robot with reinforcement learning. *Mobile robots: new research*, New York: Nova. ISBN: 1-59454-359-3. 291–315.

Erden, M. S.; Leblebicioğlu, K. 2007. Analysis of wave gaits for energy efficiency. *Autonomous Robots*. 23(3): 213–230.

Estremera, J.; de Santos, P.G. 2005. Generating continuous free crab gaits for quadruped robots on irregular terrain. *IEEE Transactions on Robotics*. 21(6): 1067–1076.

Ettlin, A. and Bleuler, H. 2006. Rough-Terrain Robot Motion Planning based on Obstacleness. *Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision*: 1–6.

Germanas, Š.; Narvydas, G. 2009. Map drawing of unknown environment using autonomous mobile robot. *The 4th International Conference on Electrical and Control Technologies*. 31–35.

Giguere, P.; Dudek, G. 2009. Clustering sensor data for autonomous terrain identification using time-dependency. *Autonomous Robots* 26(2-3): 171–186.

Gonzalez, J. P.; Dodson, W.; Dean, R.; Kreafle, G.; Lacaze, A.; Sapronov, L.; Childers, M. 2009. Using RIVET for parametric analysis of robotic systems. *In Ground vehicle systems engineering and technology symposium*.

Görner, M.; Stelzer, A. 2013. A leg proprioception based 6 DOF odometry for statically stable walking robots. *Autonomous Robots*. 34(4): 311–326.

Guizilini, V.; Ramos, F. 2013. Semi-parametric learning for visual odometry. T*he International Journal of Robotics Research* 32(5): 526–546.

Haynes, GC.; Rizzi, AA.; Koditschek, DE. 2012. Multistable phase regulation for robust steady and transitional legged gaits. *The International Journal of Robotics Research*. 31(14): 1712-1738.

Hidekazu, S.; Hitoshi, N. 2010. Quadrupedal Gait Generation Based on Human Feeling for Animal Type Robot. *Climbing and Walking Robots*. 265–276.

Hirose, S.; Lazarenko, E.; Endo, G. 2013. Non-Tumbling Gait and directional normalized energy stability margin. *Advanced Robotics* 27(15): 1137–1145.

Hirschmüller, H.; Innocent, P. R.; Garibaldi, J. 2002. Real-Time Correlation-Based Stereo Vision with Reduced Border Errors. *International Journal of Computer Vision* 47(1-3): 229–246.

Hirschmüller, H. 2008. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(2):328–341.

Hooper, R. 2009. Robot inverse kinematics. [Interaktyvus]. [Žiūrėta: 2013-06-18] http://www.learnaboutrobots.com/inverseKinematics.htm.

Hornung, A.; Bennewitz, M.; Strasdat, H. 2010. Efficient vision-based navigation. *Autonomous Robots* 29(2): 137–149.

Howard, A.; and Seraji, H. 2001. Vision-based terrain characterization and traversability assessment. *Journal of Robotic Systems* 18: 577–587.

Ishigami, G.; Nagatani, K.; Yoshida, K. 2011. Path Planning and Evaluation for Planetary Rovers Based on Dynamic Mobility Index. *IEEE/RSJ International Conference on Intelligent Robots and Systems*: 601–606.

Yunde, J.; Mingxiang, L.; Luping, A.; Xiaoxun, Z. 2003. Autonomous navigation of a miniature mobile robot using real-time trinocular stereo machine. *Processing of the IEEE International Conference on Robotics, Intelligent Systems and Signal* 1: 417–421.

Jazar, R. N. 2010. *Theory of Applied Robotics: Kinematics, Dynamics, and Control (2nd Edition)*. Springer. ISBN 978-1-4419-2. 883 p.

Jones, H.; Vaughan, R. 2010. *Remote Sensing of Vegetation: Principles, Techniques, and Applications*. Oxford University Press, 400 p.

Juang, C. F.; Chang, Y. C.; Hsiao, C. M. 2011. Evolving Gaits of a Hexapod Robot by Recurrent Neural Networks With Symbiotic Species-Based Particle Swarm Optimization. *IEEE Transactions on Industrial Electronics* 58(7):3110–3119.

Julier, S. J.; Uhlmann, J. K. 1997. A new extension of the Kalman Filter to nonlinear systems. *International Symposium on Aerospace/Defense Sensing, Simulate and Controls*: 182–193.

Kajita, S.; Tani, K. 1996. Adaptive Gait Control of a Biped Robot Based on Realtime Sensing of the Ground Profile. *Autonomous Robots* 4(3): 297–305.

Kale, A.; Salunke, G.; Kadam, K.; Jagdale, M. 2013. CPGs Inspired Adaptive Locomotion Control for Hexapod Robot. *International Journal of Engineering Science Invention* 2(4): 13–18.

Kar, D. C.; Issac, K. K.; Jayarajan, K. 2003. Gaits and Energetics in Terrestrial Locomotion. *Mechanism and Machine Theory*. 38(3): 355–366.

Koenig, N.; Howard, A. 2004. Design and use paradigms for gazebo, an open-source multi-robot Simulator. *In IEEE/RSJ International Conference on Intelligent Robots and Systems* 3: 2149–2154.

Kolter, J.Z.; Youngjun, K.; Ng, A.Y. 2009. Stereo vision and terrain modeling for quadruped robots. *IEEE International Conference on Robotics and Automation*: 1557–1564.

Kreslavsky, M. A.; Head III, J. W. 1999. Kilometer-scale slopes on Mars and their correlation with geologic units - Initial results from Mars Orbiter Laster Altimeter (MOLA) data. *Journal of Geophysical Research* 104(9): 21911–21924.

Lakaemper, R. 2004. 3D Fractal Landscapes using the Diamond Square Algorithm. [Interaktyvus]. [Žiūrėta 2013-08-31]: http://knight.temple.edu/~lakamper/courses/cis350_2004/assignments/assignment_02.htm.

Larson, A. C.;Demir, G. K.; Voyles, R. M. 2005. Terrain Classification Using Weakly-Structured Vehicle/Terrain Interaction. *Autonomous Robots* 19(1): 41–52.

Latif, Y.; Cadena, C.; Neira, J. 2013. Robust loop closing over time for pose graph SLAM. *The International Journal of Robotics Research*: 1–16.

Lengvinis, P.; Sinkevičius, S.; Simutis, R. 2011. Optical flow versus hierarchical temporal memory algorithms for human traffic monitoring system. *The 6th International Conference on Electrical and Control Technologies*. 67–70.

Lewis, M.A. 2002. Detecting surface features during locomotion using optic flow. *In IEEE International Conference on Robotics and Automation* 1: 305–310.

Manduchi, R.; Castano, A.; Talukder, A.; Matthies. L. 2004. Obstacle Detection and Terrain Classification for Autonomous Off-road Navigation. *Autonomous Robots* 18: 81–102.

Manoiu-Olaru, S. and Nitulescu, M. and Stoian, V. 2011. Hexapod robot. Mathematical support for modeling and control. *15th International Conference on System Theory, Control, and Computing (ICSTCC)*. 1-6.

Manoiu-Olaru, S.; Nitulescu, M. 2011. Hexapod robot. Virtual models for preliminary studies. *15th International Conference on System Theory, Control, and Computing (ICSTCC)*. 1–6.

Milford, M. 2013. Vision-based place recognition: how low can you go? *The International Journal of Robotics Research* 32(7): 766–789.

Mitsunaga, N.; Asada, M. 2006. How a mobile robot selects landmarks to make a decision based on an information criterion. *Autonomous Robots* 21(1): 3–14.

Molino, V.; Madhavan, R.; Messina, E.; Downs, A.; Balakirsky, S.; Jacoff, A. 2007. Traversability metrics for rough terrain applied to repeatable test methods. *In IEEE/RSJ International Conference on Intelligent Robots and Systems*: 1787–1794.

Netto, S. M. C.; Evsukoff, A.; Suell Dutra, M. 2000. Fuzzy systems to solve inverse kinematics problem in robots control: application to an hexapod robots' leg. *Sixth Brazilian Symposium on Neural Networks*. 150–155.

Olson, E.; Agarwal, P. 2013. Inference on networks of mixtures for robust robot mapping. *The International Journal of Robotics Research* 32(7): 826–840.

Ostrowski, J. P.; Desai, J. P.; Kumar, V. 2000. Optimal Gait Selection for Nonholonomic Locomotion Systems. *The International Journal of Robotics Research* 19(3): 225–237.

Petrov, P.; Dimitrov, L. 2010. Leg Trajectory Simulation for Curvlinear Motion of a Six-legged Robot. *XIX ННТК с международно участие „АДП-2010”*. 268–273.

Pfingsthorn, M.; Birk, A. 2013. Simultaneous Localization and Mapping (SLAM) with Multimodal Probability Distributions. *The International Journal of Robotics Research* 32(2): 143–171.

Premebida, C.; Nunes, U. 2013. Fusing LIDAR, camera and semantic information: A context-based approach for pedestrian detection. *The International Journal of Robotics Research* 32(2): 371–384.

Rekleitis, I.; Bedwani, J. L.; Dupuis, E.; Lamarche, T.; Allard, P. 2013. Autonomous over-the-horizon navigation using LIDAR data. *Autonomous Robots* 34(1-2): 1–18.

Ridderström, C. 2003. *Legged locomotion:Balance, control and tools – from equation to action*. Daktaro disertacija. Royal Institute of Technology. Sweden. 2003. 266 p.

Seraji, H.; Howard, A. M.; Tunstel, E. 2001. Terrain-Based Navigation of Planetary Rovers: A Fuzzy Logic Approach. *Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-Sairas)*. 1–8.

Seraji, H.; Bon, B. 2002. Multi-range traversability indices for terrain-based navigation. *In IEEE International Conference on Robotics and Automation* 3. 2674–2681.

Shepard, M. K.; Campbell, M. H.; Bulmer, M. H.; Farr, T. G.; Gaddis, L. R.; Plaut, J. J. 2001. The roughness of natural terrain: A planetary and remote sensing perspective. *Journal of Geophysical Research* 106(12): 32777–32795.

Siciliano, B.; Khatib, O. 2008. *Springer Hnadbook of Robotics*. Springer. ISBN 978-3-540-23957-4. 1611 p.

Silva, M. F.; Machado, J. A. T.; Lopes, A. M.; Tar, J. K. 2004. Gait selection for quadruped and hexapod walking systems. *Second IEEE International Conference on Computational Cybernetics*. 217–222.

Sorin, M. O.; Mircea, N. 2012. Hexapod robot locomotion over typical obstacles. *IEEE International Conference on Automation Quality and Testing Robotics (AQTR)*. 422-427.

Stavens, D; Thrun, S. 2006. A Self-Supervised Terrain Roughness Estimator for Off-Road Autonomous Driving. *In Proceedings of Conference on Uncertainty in AI*: 13–16.

Stelzer, A.; Hirschmüller, H.; Görner, M. 2012. Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. *The International Journal of Robotics Research* 31(4): 381–402.

Terrain Classification. [Interaktyvus]. [Žiūrėta: 2013-06-18] http://www.biomassenergycentre.org.uk/pls/portal/docs/page/bec_technical/sources%20of%20biomass/forestry%20for%20woodfuel%20and%20timber/harvesting/managing%20harvesting/fc-tdb-tn-16-95terrainclassification.pdf.

Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; Lau, K.; Oakley, C.; Palatucci, M.; Pratt, V.; Stang, P.; Strohband, S.; Dupont, C.; Jendrossek, L. E.; Koelen, C.; Markey, C.; Rummel, C.; van Niekerk, J.; Jensen, E.; Alessandrini, P.; Bradski, G.; Davies, B.; Ettinger,

S.; Kaehler, A.; Nefian, A.; Mahoney, P. 2006. Winning the DARPA Grand Challenge. *Journal of Field Robotics*: 661–692.

Tong, C. H.; Furgale, P.; barfoot, T. D. 2013. Gaussian Process Gauss–Newton for non-parametric simultaneous localization and mapping. *The International Journal of Robotics Research* 32(5): 507–525.

Tuley, J.; Vandapel, N.; Hebert, M. 2005. Analysis and removal of artifacts in 3-D LADAR data. *IEEE International Conference on Robotics and Automation*: 2203–2210.

Turcotte, D. L. 1997. *Fractals and Chaos in Geology and Geophysics*. Cambridge University Press, 398 p.

Umm-e-Habiba; Asghar, S. 2009. A survey on multi-criteria decision making approaches. *International Conference on Emerging Technologies*. 321–325.

Walas, K.; Belter, D.; Kasinski, A. 2008. Control and Environment Sensing System for a Six-legged Robot. *Journal of Automation, Mobile Robotics & Intelligent Systems* 2(3): 26–31.

Wang, Y. 2005. A novel decision grid theory for dynamic decision making. *Fourth IEEE Conference on Cognitive Informatics*. 308–314.

Zhang, X.; Zheng, H. 2008. Walking up and down hill with a biologically-inspired postural reflex in a quadrupedal robot. *Autonomous Robots*. 25(1–2): 15–24.

Zielinska,T. 2005. Control and navigation aspects of a group of walking robots. *Robotica* 24(01): 23–29.

Autoriaus mokslinių publikacijų disertacijos tema sąrašas

Straipsniai recenzuojamuose mokslo žurnaluose

Luneckas, T. 2010. Šešiakojo roboto eisenos tyrimas. *Mokslas – Lietuvos ateitis=Science – future of Lithuania: Elektronika ir elektrotechnika=electronics and electrical engineering*. T. 2, Nr. 1: 36–39. Vilnius: Technika. ISSN 2029-2341.

Luneckas, T. 2011. Paviršiaus netolygumo vertinimas pagal roboto padėtį. *Mokslas – Lietuvos ateitis=Science – future of Lithuania: Elektronika ir elektrotechnika=electronics and electrical engineering*. T. 3, Nr. 1: 96-99 Vilnius: Technika. ISSN 2029-2341.

Luneckas, T. 2012. Roboto eisenos parinkimas pagal pėdų koordinačių vidutinę standartinę nuokrypą. *Mokslas – Lietuvos ateitis=Science – future of Lithuania: Elektronika ir elektrotechnika=electronics and electrical engineering*. T. 4, Nr. 1: 47–50. Vilnius: Technika. ISSN 2029-2341.

Luneckas, T.; Udris, D. 2013. Terrain evaluation method for hexapod robot. *World Academy of Science, Engineering and Technology*. Vol. 7, Nr. 1. Zurich. 59–62. ISSN 2010-376X.

Straipsniai kituose leidiniuose

Luneckas, T.; Udris, D. 2009. Research of hexapod robot motion in irregular terrain. *In 5th International conference on mechatronic systems and materials MSM 2009*. Stafa-Zurich: Trans Tech Publications Ltd. 431–434. ISSN 1012-0394.

Luneckas, T.; Udris, D. 2010. The simplified hexapod robot control metod. *In proceedings of the 5th international conference on electrical and control technologies ECT-2010*. Kaunas: Technologija. 37–40. ISSN 1822-5934.

Luneckas, T.; Udris, D. 2011. Hexapod robot motion simulation. *In proceedings of the 6th international conference on electrical and control technologies ECT-2011*. Kaunas: Technologija. 31–34. ISSN 1822-5934.

Luneckas, T.; Udris, D. 2012. Hexapod kinematic model optimization for control system. *In proceedings of the 7th international conference on electrical and control technologies ECT-2012I*. Kaunas: Technologija. 113–116. ISSN 1822-5934.

Luneckas, M.; Luneckas, T.; Udris, D. 2013. Hexapod walking robot energy consumption dependence on the number of legs set on the surface. *In proceedings of the 8th international conference on electrical and control technologies ECT-2013*. Kaunas: Technologija. 25–28. ISSN 1822-5934.

Tomas LUNECKAS

ŠEŠIAKOJO ROBOTO JUDĖJIMO
NELYGIU PAVIRŠIUMI TYRIMAS

Daktaro disertacija

Technologijos mokslai,
elektros ir elektronikos inžinerija (01T)

ANALYSES OF HEXAPOD ROBOT
MOVEMENT OVER ROUGH TERRAIN

Doctoral Dissertation

Technological Sciences,
Electrical and Electronic Engineering (01T)

2013 11 18. 8,5 sp. l. Tiražas 20 egz.
Vilniaus Gedimino technikos universiteto
leidykla „Technika“,
Saulėtekio al. 11, 10223 Vilnius,
http://leidykla.vgtu.lt
Spausdino UAB „Ciklonas“
J. Jasinskio g. 15, 01111 Vilnius