VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
KOMPIUTERIJOS KATEDRA

Baigiamasis magistro darbas

Nežinomų teritorijų tyrinėjimas naudojant savaeigius robotizuotus mechanizmus

Atliko: 2 kurso studentas
Stanislav Zachaževski (parašas)

Darbo vadovas:
dr. Alminas Čivilis (parašas)

Vilnius
2009

Turinys

1 Apibrėžimai

- NTD – nežinomų teritorijų dengimas (angl. *Robotic area coverage or covering salesman problem*);
- JP – judėjimo planavimas (angl. *Motion planning algorithms*);
- ASK – analoginis skaitmeninis keitiklis;
- CPU – centrinis procesorius (angl. *Central processor unit*).

2 Anotacija

Nežinomo ploto dengimas yra aktuali ir paplitusi problema. NPD sprendimas realiuose robotuose susiduria su daviklių ir mechanizmų netikslumu. Atliktame darbe yra pateiktas „Bouncing“ NPD algoritmo sprendimas robotui, turinčiam mažo tikslumo daviklius ir neprecizinius valdiklius. Taip pat atliktas darbas parodė sudėtingus roboto kūrimo aspektus ir galimus sprendimus. Sukurtas robotas dėl pigumo ir nesudėtingos realizacijos gali būti naudojamas kaip platforma kitokių algoritmų tyrimui.

3 Summary

The problem of unknown area coverage with mobile robots has received considerable attention over the past years. This problem is a common challenge in many applications, including automatic lawn mowing and vacuum cleaning. However, most of the approaches find difficult to implement in real life because of problems of environment data reading. In this paper we consider the problem of robust area covering algorithm implementation in mobile robot. The chosen approach is based on simple and robust algorithm for uncertain environment and simple robot platform. The results showed robustness, reliability of chosen method of control. The constructed robot has shown simplicity, cheapness of creation and possibility for different algorithm testing. The significance of this paper lies in the practical solution for robust mobile robot area coverage, suitable for noisy environment and low precisions robot sensors.

4 Įvadas

Nežinomų teritorijų dengimas (NTD) (angl. *Robotic area coverage or covering salesman problem*) yra aktuali ir paplitusi tema. NTD nustato kelią, kuriuo einant bus optimaliai padengtas ištisas teritorijos plotas. Jos taikymas yra labai platus: nuo dulkių siurblio iki robotų - minų ieškotojų. Teritorijų dengimo tematika priklauso judėjimo planavimo (JP) (angl. *Motion planning algorithms*) problemų aibei [Lav06]. Savo ruožtu JP raidos metu apjungė tris atskiras mokslo šakas: robotiką, dirbtinį intelektą ir valdymo teoriją (angl. *Robotic, artificial intelligence and control theory*) [Lav06]. JP problematikoje jau atsirado ir taip vadinami standartiniai uždaviniai. Vienas iš populiariausių yra labirinto apėjimas (angl. *Maze solving*) arba keliaujančio pirklio problema (angl. *Travelling salesman problem*) [GP06]. Mažiau žinomas reiškinys – NTD pradėjo sparčiai vystytis roboto technikai išėjus už laboratorijų ribų.

Kaip jau buvo minėta, NTD problematika sprendžia dažnai sutinkamą realaus gyvenimo problemą. Greta minėtų dulkių siurblių ir robotų - minų ieškotojų galima būtų paminėti ir kitus NTD naudojimo atvejus: įvairūs žemės apdirbimo darbai (žolės pjovimas, žemės arimas), dažymas, cheminis apdirbimas, žmonių paieška užgriuvusiose uolose/šachtose arba namo griuvėsiuose po žemės drebėjimo. Taip pat galima paminėti ir nežinomų teritorijų autonominę žvalgybą [Cho01], [MJ04], [CP97], [Cho00]. Dauguma šių įrenginių jau pradėti gaminti masiškai. Autoriui yra žinomi pramoniniu būdu gaminami skirtingų gamintojų robotizuoti dulkių siurbliai, robotizuota vėjapjovė, baseino valymo robotai.

5 Susiję darbai

NPD problematika dažnai sutinkama realaus gyvenimo uždaviniuose. Sprendimai, tinkantys nepatikimiems daviklių rodmenims, yra aprašomi šaltiniuose [MJ04], [AC01]. Autoriai pateikia keletą sprendimų, tinkančių robotams su nesudėtingais jutiklias. Darbe [MJ04] pateikiami skirtingų valdymo algoritmų tyrimai, priklausomai nuo roboto daviklių paklaidos. Buvo nauodojami tokie algoritmai:

- nominalus valdymas (angl. *Nominal control*);
- garantuojantysis valdymas (angl. *Guaranteed control*);
- galimas valdymas (angl. *Possible control*);
- euristinis valdymas (angl. *Heuristic control*);
- atsitiktinis valdymas (angl. *Randomized control*).

Tyrimų rezultatai parodė mažėjantį sudėtingų algoritmų pranašumą, lyginant su paprastais algoritmais, kai roboto daviklių paklaida didėja. Taip pat tyrimai rodo, kad vieni sprendimai reikialauja daugiau skaičiuojamosios galios ir užtikrina tikslesnį teritorijos apėjimą, kiti atvirkščiai – jiems atlikti reikia mažiau skaičiavimų, tačiau optimalus dengimas neužtikrinamas.

Darbo [AC01] autoriai pateikia mažai resursų reikalaujantį algoritmą, tinkantį atpažinti nekorektiškus daviklio signalus. Atlikti tyrimai ir testavimas naudojant tikrą robotą parodė gerus rezultatus. Algoritmas užtikrino nežinomos teritorijos dengimą, esant roboto daviklių trukdžiams.

Dinaminiai dengimo algortimai reikalauja optimalaus kelio radimo žinant dalį dengiamo ploto žemėlapio. Tokie atvejai aprašomi darbuose [CP97], [Cho00]. Problemai spręsti autoriai pateikia „Boustrophedon Cellular Decomposition“ algoritmą. Algoritmas užtikrina optimalų ir visišką ploto su kliūtimis apėjimą, palyginus su įprastiniu „Boustrophedon“ tipo algoritmu. Taip pat pažymima, kad optimaliam rezultatui pasiekti algoritmui būti korektiški duomenys.

Įdomi tematika yra kelių robotų to paties ploto tyrinėjimas – dengimas. Tokia problematika yra aprašoma šaltinyje [BRH00]. Autorių pasiūlytas „DCr“ algoritmas leidžia optimaliau kooperuoti kelių robotų darbą. Pasiekti rezultatai parodė, kad, naudojant tokį algoritmą, kai dalyvauja du robotai ir kiekvienas jų dengia pusę visos teritorijos, nėra dubliuojamas to paties ploto dengimas. Tokiu budu užtikrinamas optimalus ploto dengimas.

Išnagrinėjus kitų autorių darbus, galima pastebėti, kad, norint tinkamai išnaudoti sudėtingų algoritmų siūlomus pranašumus, reikia korektiškų daviklių signalų. Jei daviklių signalai nėra korektiški, sudėtingi algoritmai nepasiteisina – algoritmo efektyvumas mažėja. Taip

pat autoriui žinomi komerciniai robotai, skirti spręsti uždavinius su NPD, naudojantys įvarias „Bouncing“ tipo algoritmo modifikacijas. Taigi norint užtikrinti roboto patikimumą, reikia naudoti paprastus algoritmus, o naudojant sudėtingus – palikti galimybę naudoti paprastesnį algoritmą, jei davikliai sugestų arba atsirastų daug trukdžių.

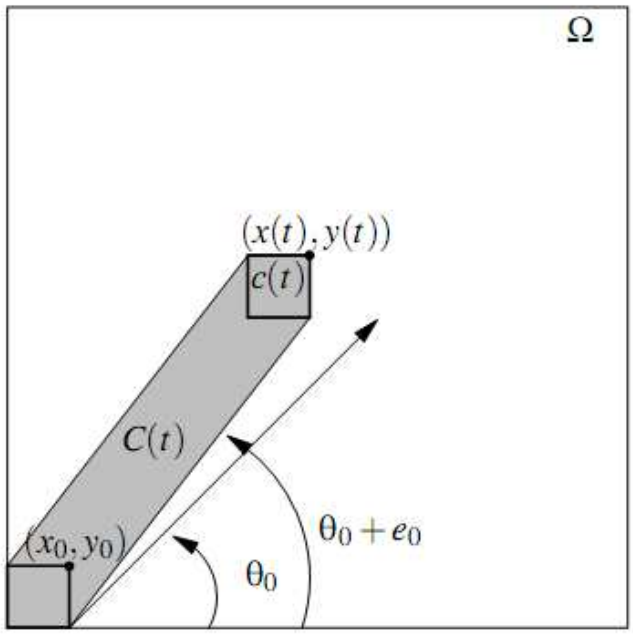
6 Problemos aprašymas ir galimi problemos sprendimai.

6.1 Problemos formulavimas

Sprendžiame tokią aibės dengimo problemą:

$$\Omega = [0,L] \times [0,L] \subset \mathbb{R}^2, \quad L > 1,$$

Naudojant stačiakampį robotą, kaip pavaizduota pav. 1, robotas dengia ploto vienetą, esantį koordinatėse (x,y):



Pav. 1 Stačiakampio roboto ploto dengimas

Laiku
$t \geq 0$, robotas dengia aibę:

$$c(t) = [x(t) - 1, x(t)] \times [y(t) - 1, y(t)].$$

Visą padengtą aibę galima pažymėti:

$$C(t) = \bigcup_{s \in [0,t]} c(s).$$

Diskretizuojant laiką ir įvedant judėjimo žingsnio sąvoką, vienam žingsniui trunkant t, viso ploto dengimas geriausiu atveju užtruktų:

$t = S/s$, kur S – dengiamo ploto plotas, s – roboto dengiamas plotas vieno žingsnio metu.

Paprasčiausiai realiai sistemai robotas papildomai sugaišta laiko posūkiams:

$t = S/s + nt_{(pos)}$, kur papildomai n – posūkių skaičius, t(pos) – posūkių trukmės laikas.

Norint optimizuoti ploto dengimą realioje sistemoje, reikia sumažinti posūkių skaičių. Mažinti dengimo žingsnių skaičiaus negalime – liks nepadengto ploto.

Imant realų uždavinį – ploto dengimą ir veiksmo atlikimą kiekvieno dengiamo ploto taške, robotui - žoliapjovei reikia papildomo laiko veiksmui atlikti. Ta pati žolė neauga tolygiai visoje pievoje. Pjovimas duotajame ploto taške gali reikalauti skirtingo laiko, priklausomai nuo žolės aukščio. Taip pat roboto judėjimo posistemė gali turėti blogą sukibimą su danga. Roboto faktinė judėjimo trajektorija skirsis nuo suplanuotos – atsiras roboto judėjimo paklaida. Papildomai gali įvykti dalinis daviklių sistemos gedimas, davikliai teiks nekorektiškus duomenis. Papildomai optimizuojant NPD sprendimą, galima naudoti dinaminius algoritmus – robotas kuria padengto ploto žemėlapį [CK99], pagal kurį optimizuoja dengimo algoritmą.

Esant realiai NPD problemos implementacijai, algoritmas sudėtingėja. Norint tinkamai spręsti NPD uždavinį, šalia teorinio uždavinio sprendimo dar reikia tikrinti daviklių signalų korektiškumą, faktinę/suplanuotą teritoriją, stebėti sistemos gedimus ir juos kompensuoti.

6.2 NPD algoritmai skirti realiems robotams

NPD problematikai spręsti yra naudojami įvairūs algoritmai. Bendru atveju algoritmus galima padalinti į dvi grupes. Pirmajai grupei priklauso statiniai algoritmai, tai yra algoritmai, kurie nenaudoja dengiamo ploto žemėlapio (angl. *Non-map building autonomous robot*). Antrajai grupei priklauso dinaminiai algoritmai, kuriantys dengiamo ploto žemėlapius (angl. *Map building autonomous robot*) [CK99]. Dinaminiai algoritmai turi tokius pranašumus:

- leidžia optimaliau planuoti apėjimo trajektoriją;
- prisitaiko prie naujų kliūčių;
- garantuoja viso ploto dengimą.

Dinaminio algoritmo trūkumai būtų tokie:

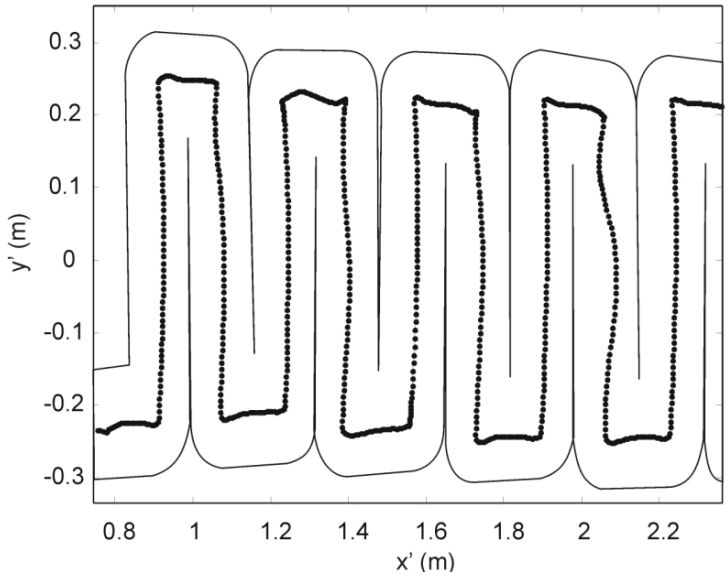
- tikslesnių įvesties duomenų reikalavimas;
- didesnis skaičiuojamosios galios poreikis;
- mažesnis sistemos patikimumas.

Savo ruožtu, statiniai algoritmai turi tik tokį trūkumą, kad neužtikrina optimalaus ploto dengimo. O jų privalumai būtų tokie:

- patikimumas;
- paprasta realizacija.

Taip pat algoritmas privalo prisitaikyti prie realaus NPD veikimo aplinkos – daviklių signalo nekorektiškumo, atskirų roboto sistemų gedimų. Galima pateikti analogiją su žmogumi.

Paprastai žmogus gauna apie septyniasdešimt procentų informacijos vizualiai – akimis. Didesnė žmogaus smegenų dalis yra skirta vaizdinės informacijos apdorojimui. Bet žmogui praradus regėjimą, smegenys geba „perjungti“ skaičiuojamąją galią iš vaizdų apdorojimo į klausos, uoslės ir jutimo. Tai pilnai nepakeičia regėjimo organų, bet leidžia žmogui tiksliau orientuotis erdvėje, panaudojant „laisvąją skaičiuojamąją galią“. Pagal pateiktą pavyzdį galima matyti, kaip gamta stengiasi efektyviai išnaudoti visus įmanomus resursus, kad žmogus išgyventų. Panašių situacijų analizė yra pateikta [MJ04], [AC01]. Darbo [MJ04] autoriai aprašo algoritmus, tinkančius tiems robotams, kurie yra patyrę jutiklių pažeidimus. Pateikti algoritmai naudoja pačius primityviausius, t.y. jutimo, daviklius. Autorių pateikta algoritmų analizė leidžia optimaliai dengti nežinomą teritoriją, naudojant tik jutimo daviklius su didele paklaida. Pateikta analizė įrodo, kad euristinis algoritmas leidžia apeiti plotą su minimaliu posūkių ir persidengimų kiekiu. Algoritmas naudoja paprastą „Boustrophedon“ kelią, kuris aprašo judėjimą pirmyn - atgal su minimaliais persidengimais (pav. 2):

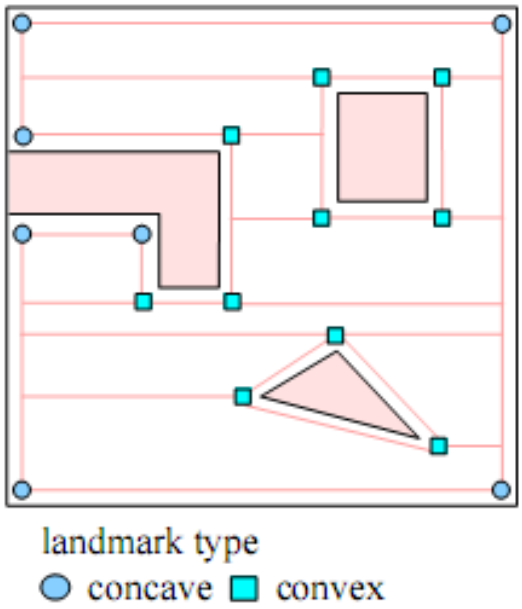


Pav. 2 „Boustrophedon“ kelias

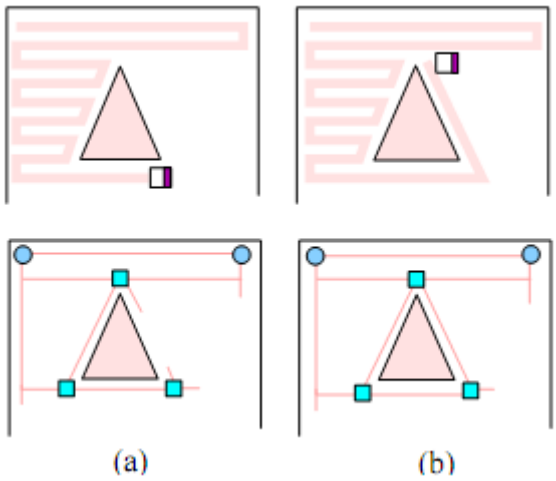
Taip pat yra pateikti ir kiti algoritmai, bet jie yra kompromisiniai. Tai yra, arba kenčia dengimo kokybė, arba didėja reikiamų resursų skaičius, įvesties duomenų tikslumas, norint optimaliai padengti plotą.

Naudojant dinaminį algoritmą realioje sistemoje, iškyla žemėlapio duomenų apdorojimo ir saugojimo problema. Kaip išeitis yra siūloma naudoti žymes [WCM00]. Darbe pateiktas

euristinis „Boustrophedon“ algoritmas naudojant žymes, kurių pagalba žymimos dengiamo ploto kliūtys.



Pav. 3 Žymių naudojimas, kuriant nežinomo ploto žemėlapį



Pav. 4 Apėjimo pritaikymas naujai kliūčiai (b)

Autorių pateiktas sprendimas garantuoja visišką ploto dengimą.

Atlikta realaus NPD uždavinio problematikos analizė parodė problemos netrivialumą. Taip pat ir tai, kad nėra universalaus algoritmo, tinkančio visiems atvejams. Dauguma optimalių algoritmų yra reiklūs įvesties informacijai, reikalauja daug resursų. Esant netiksliems įvesties duomenims, sudėtingų algoritmų optimalumas sumažėja. Taigi galime pastebėti, kad, norint sukurti robotą, skirtą spręsti realų NPD uždavinį, reikia išspręsti daug papildomų problemų:

- realių duomenų nekorektiškumo;
- darbinių mechanizmų paklaidos;
- posistemių gedimo.

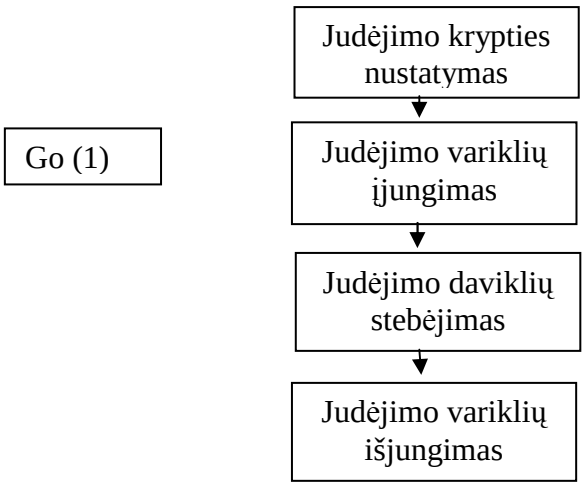
Atliekamo tyrimo tikslas bus atskliesti, kokios problemos iškyla kuriant robotą, skirtą spręsti realią NPD problemą.

7 Tyrimas

7.1 Bendri NPD problemos tyrimo aspektai

Pagrindinis NPD uždavinys – rasti optimalų ploto dengimo algoritmą, kurio pagalba robotas optimaliai padengtų visą plotą. Algoritmas turėtų būti patikimas. Todėl jis privalo būti atsparus daviklių posistemės trukdžiams ir gedimams. Visiems šiems dalykams užtikrinti reikia naudoti paprastus algoritmus, tokius kaip „Bouncing“ tipo algoritmas. Sudėtingesni algoritmai neužtikrina optimalesnio dengimo, bet padidina sistemos sudėtingumą ir sumažina patikimumą. NPD algoritmo testavimui yra naudojami du skirtingi būdai: realus roboto modelis arba simuliacine aplinka. Galima būtų paminėti tokias komercines simuliavimo aplinkas: Simbad 3d Robot Simulator, Microsoft Robotic Studio. Tačiau šios simuliavimo programos leidžia ištestuoti tik algoritmo veikimą, bet realiam robotui sukurti to neužtenka.

Kaip pavyzdį galima pateikti judėjimą į priekį vieno metro atstumu. Simuliaciniame modelyje tam užtenka komandos GO(1). Realioje sistemoje po GO procedūra slepiasi sudėtingas algoritmas, kurio tikslas – duoti komandą varomajai posistemei, stebėti judėjimo daviklius bei vidines posistemes. Grafiškai tai galėtų atrodyti taip [Pav 5]:



Pav. 5 Diagrama

Iš pateikto pavyzdžio galime matyti, kad paprastos komandos įgyvendinimas kuriant realų robotą yra netrivialus uždavinys. Taip pat yra ir su sąsaja, skirta NPD algoritmo ir vidinių sistemų apjungimui.

Kito tyrimo tikslas bus parodyti galimas problemas ir jų sprendimus kuriant realų robotą, skirtą NPD užduoties sprendimui.

7.2 Realios NPD problemos algoritmo aspektai

Paprasčiausiu atveju realus robotas, skirtas spręsti NPD uždavinį, privalo atlikti tokius veiksmus:

- judėti;
- suvokti aplinką;
- atlikti užprogramuotą veiksmą;
- skaičiuoti NPD trajektoriją.

Pagal pateiktą uždavinį robotas privalo turėti tokias posistemes:

- kinematinė posistemė, skirta roboto judėjimui;
- jutiklinė posistemė, skirta aplinkos suvokimui;
- darbinė posistemė, skirta užprogramuotam veiksmui atlikti;
- skaičiuojamoji posistemė, skirta atlikti NPD algoritmo skaičiavimus.

Žemiau bus pateikti galimi sprendimai išvardintoms posistemėms bei pasirinkti tinkamiausi iš jų NPD roboto realizacijai. Norint apriboti posistemių aibę, bus paminėti sprendimai, tinkantys robotui - žoliapjovei.

8 NPD roboto posistemių specifika praktikoje – savaeigė, robotizuota žoliapjovė

Šiame skyriuje bus pateikti galimi posistemių sprendimai ir parinkti tinkamiausi robotui - žoliapjovei.

Kinematinė posistemė

Roboto kinematinė posistemė yra skirta keisti roboto judėjimą nustatyta kryptimi ir taip pat keisti pačią kryptį. Gamtoje dauguma gyvų sutvėrimų juda kojų pagalba. Todėl nuo pat robotikos pradžių yra daromi eksperimentai tokiam judėjimui atkartoti. Yra pasiekta neblogų rezultatų, bet toks sprendimas reikalauja daug resursų, yra brangus ir nepatikimas. Taip yra dėl sudėtingos kinematikos ir reikalavimo robotui „žiūrėti po kojomis“. Tai, kas žmogui atrodo natūralu, robotikoje reikalauja daug skaičiuojamosios galios ir tikslių daviklių. Dėl to pramoniniai robotai naudoja seną žmonijos išradimą – ratus. Tai yra žymiai paprastesnis sprendimas, kuriam nereikia daug skaičiuojamosios galios. Taip pat roboto judėjimui panaudojus ratus, žymiai supaprastėja roboto kinematika. Roboto judėjimui tokiu atveju paprastai užtenka vieno variklio su valdikliu, kad jis galėtų judėti į priekį. Norint keisti roboto judėjimo kryptį, galimi du variantai:

- galima valdyti ratų tarpusavio pasukimo kampą;
- turėti 2 judėjimo variklius, varančius skirtingų roboto pusių ratukus. Tokiu būdu posūkiui atlikti reikia paduoti skirtingą signalą į atskirų pusių variklių valdiklius. Toks valdymas vadinamas „diferenciniu“.

  Abu sprendimai turi savo privalumų ir trūkumų:

- valdymas, naudojant atskiro ratuko/ratukų pasukimus, turi trūkumą – reikalauja gero sukibimo su danga. Mūsų duotajame uždavinyje važiuojamoji danga yra žolė. Sausa žolė suteikia pakankamai gerą sukibimą su roboto ratais, tačiau drėgna žolė yra slidi, todėl posistemė veiktų nepatikimai. Kita vertus, tokio tipo valdymas reikalauja tik vieno varančio variklio ir yra pigesnis, lyginant su diferenciniu valdymu;
- valdymas, naudojant diferencinę pavarą, yra sąlyginai brangesnis, ir reikalauja dviejų varančių variklių, dviejų variklio valdymo blokų. Tačiau toks valdymas nėra reiklus paviršiui, suteikia robotui didesnį manevringumą, leidžia apsisukti vietoje – daryti 360° kampo posūkius, kurie nėra įmanomi naudojant rato/ratų pasukimą. Taigi kinematinė posistemė turi geresnį pravažiamumą, geresnį roboto manevringumą. Tai patvirtina ir

realaus gyvenimo atvejai. Visi robotai žvalgai, išsiųsti į svetimas planetas, buvo varomi diferencinės pavaros pagalba.

Duotam uždaviniui – robotizuotam žolės pjovimui – yra parinkta diferencinė ratų pavara. [Pav. 6] yra parodyta sukurto roboto diferencinė pavara:

Pav. 6 Sukurto roboto diferencinė pavara

Kaip varantieji mechanizmai buvo panaudoti radio bangomis valdomų modelių servo-mechanizmai. Po nedidelių modifikacijų, servo-mechanizmai buvo perdaryti į variklius, tinkančius robotui. Taip pat servo-mechanizmai turi integruotą variklio valdiklį, o tai dar labiau supaprastino variklių valdymą.

Jutiklinė - darbinė posistemė

Roboto jutiklių posistemė yra skirta aplinkos atpažinimui/jutimui. Konkrečiu roboto - žoliapjovės atveju robotas turi jausti dengiamo ploto ribas, skirtumą tarp nupjautos ir nenupjautos žolės, bei atsiradusias kliūtis. Kokie davikliai tiktų tokiam uždaviniui?

Darbinio ploto apribojimui komercinės žoliapjovės naudoja elektromagnetines bangas, kurias skleidžia kabelis. Elektromagnetinių signalų jutimui yra naudojama elektromagnetinė ritė, sujungta su signalo apdorojimo schema. Toks dengiamo ploto apribojimas yra pakankai patikimas ir nesudėtingai realizuojamas.

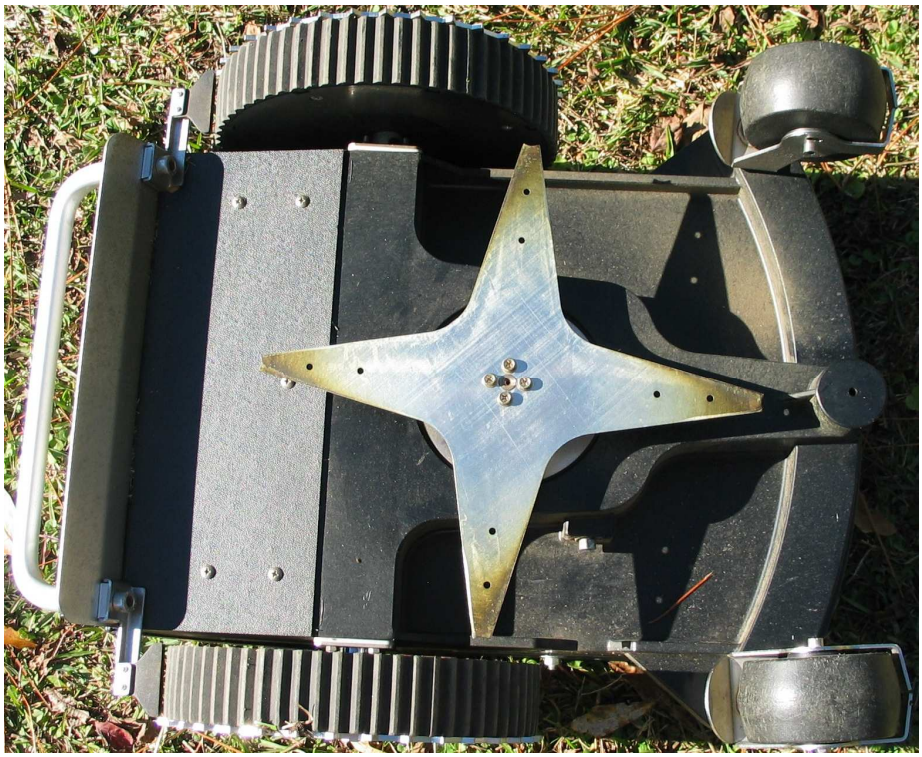
Kitas uždavinys yra pjaunamos žolės krašto stebėjimas. Tai šiek tiek pasunkina užduotį. Pjaunamos žolės kraštui stebėti galime naudoti optinius daviklius**.** Optinis daviklis stebėtų žolės aukštį, pagal kurį valdymo posistemė spręstų, ar duotajame plote reikia atlikti užprogramuotą veiksmą – nupjauti žolę. Realiu atveju optinio daviklio naudojimas sukelia tokias problemas kaip neveikimą ir teikiamų duomenų nekorektiškumą. Tai lemia aplinka, kurioje dirba davikliai.

Optinis daviklis, veikdamas robote, kuris pjauna žolę, pastoviai pasidengia plonu nupjautos žolės sluoksniu, nepriklausomai nuo jo pritvirtinimo vietos bei galimų apsaugų. Pjaunant žolę, susidaro aerozolinio tipo dulkės, kurios pasiekia visas žoliapjovės detales. Analogiška pavyzdinė situacija – pramoninis dažymas. Jo metu optinis daviklis pasidengia dažais ir tampa neveiksmingas. Pramonėje ši problema buvo išspręsta – tokie robotai yra „akli“, t.y. neturi jokių daviklių, pasakančių, kuris plotas jau padengtas dažais, kuris plotas padengtas nevisiškai arba yra neatitinkantis reikalavimų. Tokie robotai turi tik tokius daviklius, kurie nurodo dažančio instrumento padėtį erdvėje. Norint tinkamai suprogramuoti tokį robotą, programuoti tektų keletą mėnesių [Cho02], nes būtina suprogramuoti dažymo instrumento judėjimo trajektoriją, garantuojančią optimalų viso ploto dengimą. Paprastai tai yra daroma bandymo būdu.

Šaltinio [Cho02] autorius pasiūlė atvirkštinį sprendimą – analitiškai paskaičiuoti dažančio instrumento judėjimą. Tam tikslui algoritmas naudojo dengiamo ploto 3D modelį, galimas roboto rankos padėtis, dažančio instrumento dengimo plotą priklausomai nuo dažomo objekto padėties. Remiantis tokiu sėkmingu ir plačiai naudojamu pavyzdžiu, būtų galima sukurti robotą automatą, atkartojantį įrašytą, iš anksto apskaičiuotą trajektoriją. Naudojant tokį veikimo algoritmą, robotas padengtų visą plotą. Tačiau toks sprendimas tinka tik statinei žinomai aplinkai ir reikalauja labai preciziškos mechanikos. Iš pateiktos analizės galima matyti, kad nėra tokio daviklio, kuris užtikrintų patikimą roboto - žoliapjovės jutimą. Tai patvirtina ir pramoniniai robotai.

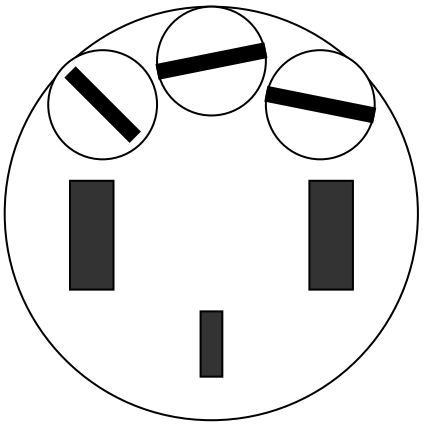
Dauguma pramoninių robotų naudoja tik pjovimo ribos daviklius, o kaip NPD algoritmą naudoja „Bouncing“ tipo algoritmą. Nagrinėjamas pramoninis robotas „Robomower“

pavaizduotas [pav. 7] turi įdomią savybę: įvažiavęs į aukštą žolę, robotas sulėtina važiavimo greitį ir pakeičia trajektoriją iš tiesios į spiralę. Spirale jis dengia aukštos žolės plotą. Tai yra padaryta siekiant apsaugoti roboto darbinį variklį nuo perkrovimo. Tam tikslui yra matuojamas darbinio variklio apkrovimas. Įvažiavus robotui į aukštą žolę, variklio apkrovimas didėja. O važiuojant virš nupjautos žolės – peiliukai sukasi laisvai, variklio apkrovimo nėra.

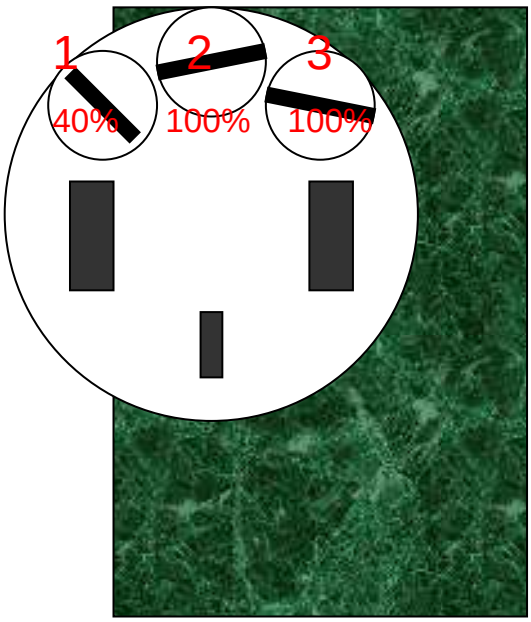


Pav. 7 Šaltinis: http://www.robomower-robotic-lawnmower.com/evolutionreview.htm

Pagal pateiktą pavyzdį galima pateikti patikimo jutiklio sprendimą robotui - žoliapjovei. Sumontavę vietoj vieno darbinio peilio [pav. 7] tris mažus [pav. 8], ir matuodami peiliukų apkrovimus [Pav. 9], gauname patikimą jutiklinę posistemę su papildomu kalibravimo davikliu.



Pav. 8 Daviklių išdėstymas



Pav. 9 Daviklių apkrovos matavimas

[Pav. 9] galima matyti daviklio rodmenis robotui judant palei žolės kraštą. Antras ir trečias dengia nenupjautą žolę, turi pilną pačią apkrovą. Pirmas peiliukas tik dalinai dengia nenupjautą žolą, todėl jo apkrovimas bus mažesnis už antro ir trečio peiliuko. Turint ir kitus duomenis, galima pateikti algoritmą, skirtą roboto judėjimui palei nupjautos žolės kraštą.

Pirmas pjovimo peiliukas dengia apie 40 proc. galimo dengimo ploto, antras ir trečias peiliukai dengia visą pjovimo plotą [Pav 9]. Algoritmas lygina pirmo ir trečio (kraštinius) daviklius su antro (pavyzdinio) daviklio parodymais. Norint robotui sekti palei pjaunamos žolės kraštą, reikia matuoti diferencinį apkrovimą tarp pirmo ir trečio daviklio. Atvirkštinis diferencinis signalas yra siunčiamas kinematinei posistemei. Antras daviklio apkrovimo lygis naudojamas kaip pavyzdinis signalas. Aprašytas algoritmas yra plačiai žinomas kaip „linijos sekimo“ algoritmas (angl. *Line following*). Analogiškai matuojant antro daviklio apkrovimą ir palyginus jį su pirmu ir trečiu, galima jausti ir pjaunamos žolės kraštą.

Tokio daviklio veikimo reikėtų „Bouncing“ algoritmui. Apjungus abu algoritmus, galima gauti tokį algoritmą, kuris optimaliai ir be pasikartojimo dengia nežinomą plotą. Taip pat toks tokių daviklių panaudojimas užtikrina patikimą darbą roboto - žoliapjovės aplinkoje. Pateiktas sprendimas turi ir papildomą pranašumą: robotas pereina į kitą dengiamo ploto vienetą tik įvykdęs užduotį esamame ploto vienete. Tokiu būdu užtikrinama užduoties atlikimo kontrolė. Analogiškas daviklių sprendimas gali būti naudojamas ir sniego valymo robotuose.

Rastas sprendimas užtikrina patikimą aplinkos jutimą ir harmoningai apjungia jutiklinę bei darbinę posistemę. Taip pat šis sprendimas garantuoja ne tik viso ploto dengimą, bet ir užprogramuotos užduoties atlikimą visame plote.

8.1 Skaičiuojamoji posistemė

Skaičiuojamoji posistemė užtikrina užprogramuoto NPD algoritmo veikimą, duomenų apdorojimą. Tam tikslui pramoniniai robotai naudoja mikroprocesorius. Pagrindinis mikroprocesorių skirtumas – skaičiuojamoji galia. Yra gaminami 4 bitų, mažo dažnio mikroprocesoriai, skirti kišeniniam skaičiuotuvui ir nepaprastai galingi, integruojantys keletą milijardų tranzistorių. Bet už skaičiuojamąją galią reikia mokėti – kišeniniam skaičiuotuvui dviejų mažų maitinimo baterijų užtenka pusmečiui, galingam CPU (angl. *Central Processor Unit*) tokio energijos kiekio neužtektų nė sekundei. Taigi ieškome sprendimo mobiliam robotui su autonominiu maitinimu. Norint rasti kompromisą – mažą energijos naudojimo lygį ir pakankamą skaičiuojamąją galią buvo pasirinkti mikrovaldikliai. Mikrovaldiklis – tai CPU, integruotas į vieną mikroschemą kartu su operatyvine, programine atmintimis ir kita reikalinga CPU veikimui periferija. Mikrovaldikliai užtikrina pilną kompiuterio funkcionalumą, būdami integruoti į vieną mikroschemą. Duomenų apsikeitimui yra naudojami išoriniai portai.

Kuriamo roboto modeliui buvo pasirinktas „Atmel“ firmos, AVR šeimos AtMega32 mikrovaldiklis. „Atmel“ įmonės mikrovaldiklio pasirinkimą lėmė šie privalumai:

- veikimo greitis (dauguma komandų yra atliekamos vieno ciklo metu, tuo tarpu konkurentai tam sugaišta keturis ciklus);
- GCC C kalbos kompiliatoriaus palaikymas (kitų įmonių mikrovaldikliams yra naudojami tik specialūs ir nepalaikantys standartizuotos C kalbos kompiliatoriai);
- prieinamumas ir mažos kainos.

Pasirinkta skaičiuojamoji posistemė yra patikrinta ir plačiai naudojama tarp robotų kūrėjų. Taip pat yra parašyta daug standartizuotų C kalbos bibliotekų.

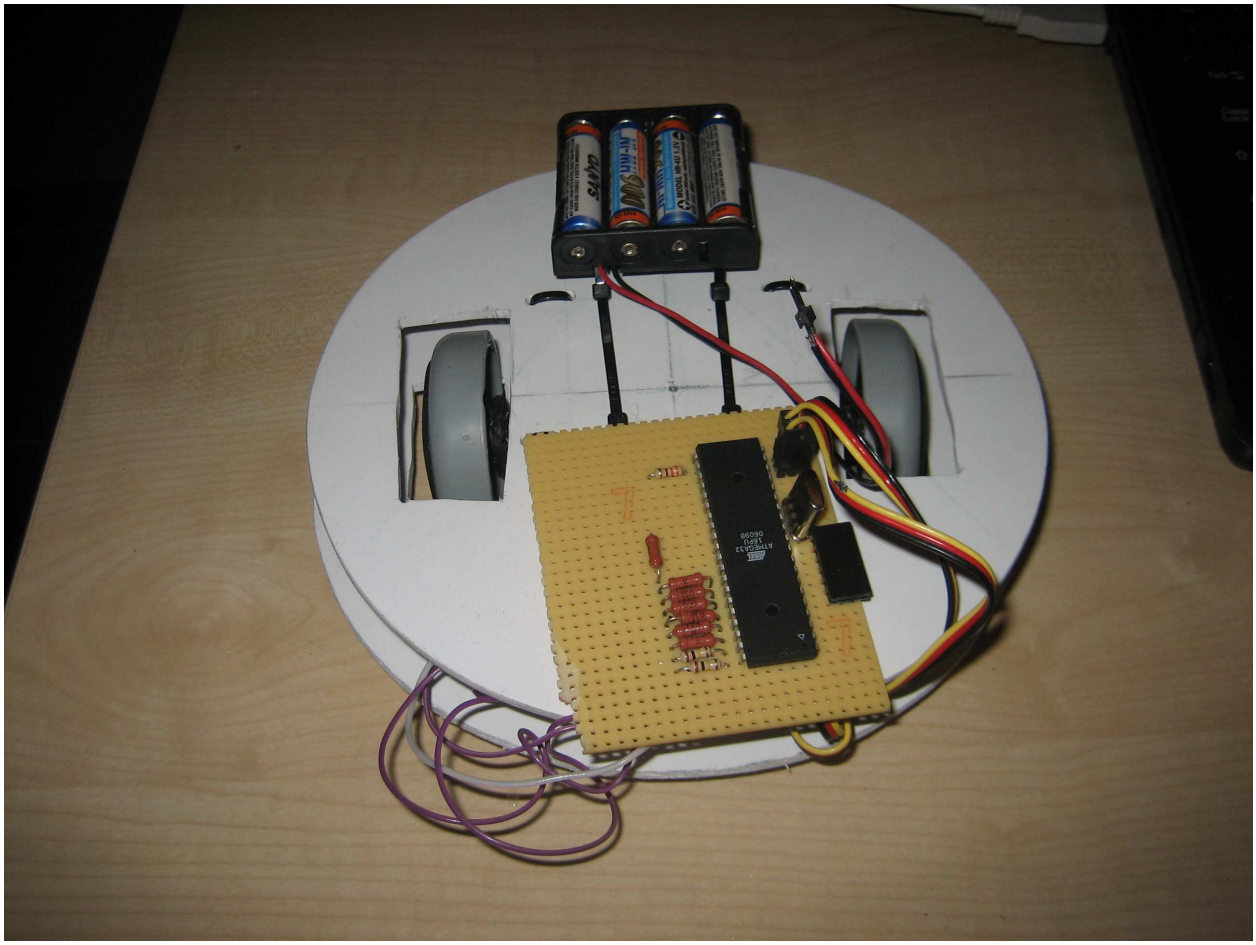
Kita plačiai naudojama posistemė – nešiojamasis kompiuteris – buvo atmesta dėl šių priežasčių:

- perteklinė skaičiuojamoji galia;
- dideli gabaritai;
- didelė kaina.

Taigi sukurtam robotui bus naudojama skaičiuojamoji posistemė AtMega32 mikrovaldiklio pagrindu.

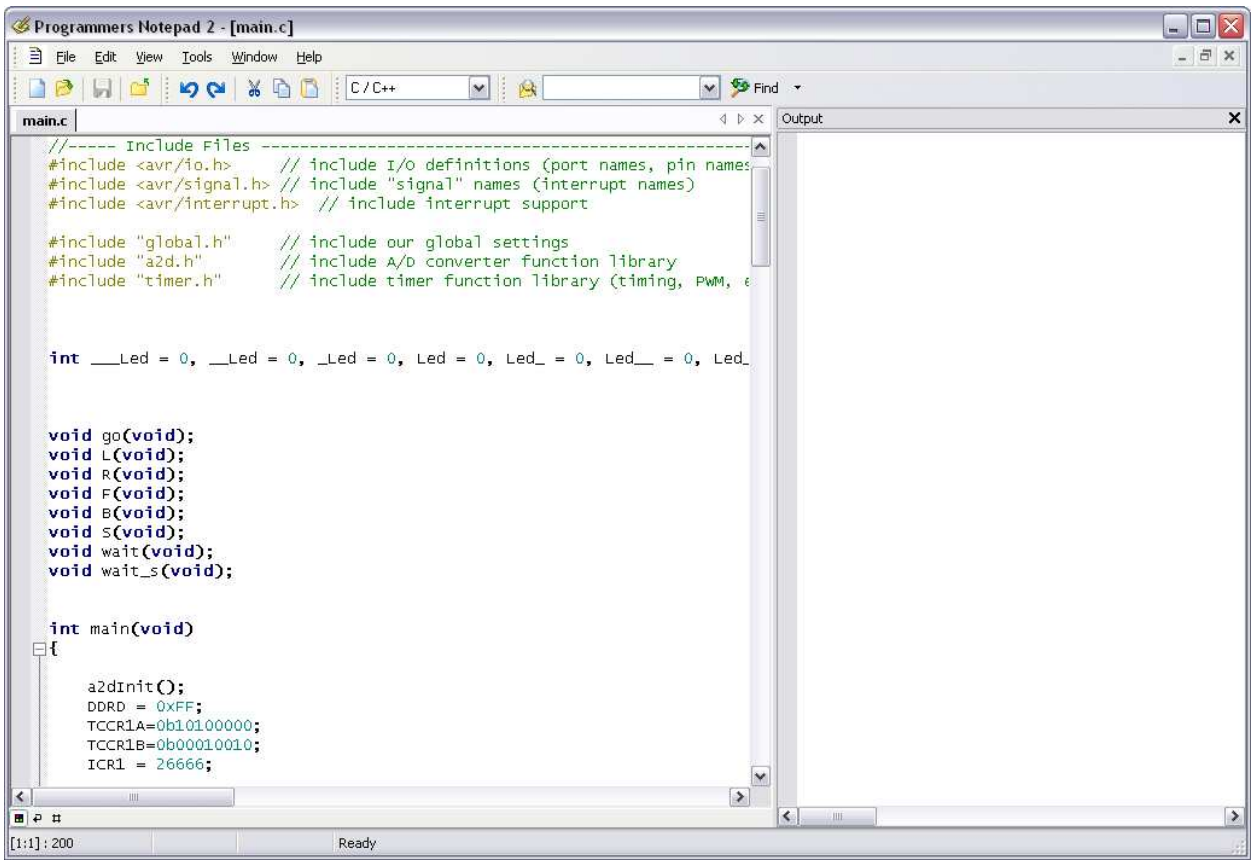
8.2 Bendra NPD algoritmo modeliavimo specifikacija

Pagal pasirinktus posistemių kriterijus buvo sukurtas roboto modelis. Roboto sandara atitinka realų robotą ir jo veikimo principą. Sukurtas modelis pavaizduotas [Pav. 10]:



Pav. 10 Sukurto roboto modelis

Mikrovaldiklio kodo rašymui buvo naudotas WinAvr GCC kompiliatorių rinkinys ir redaktorius „Programmers Notepad“ [Pav 11]:

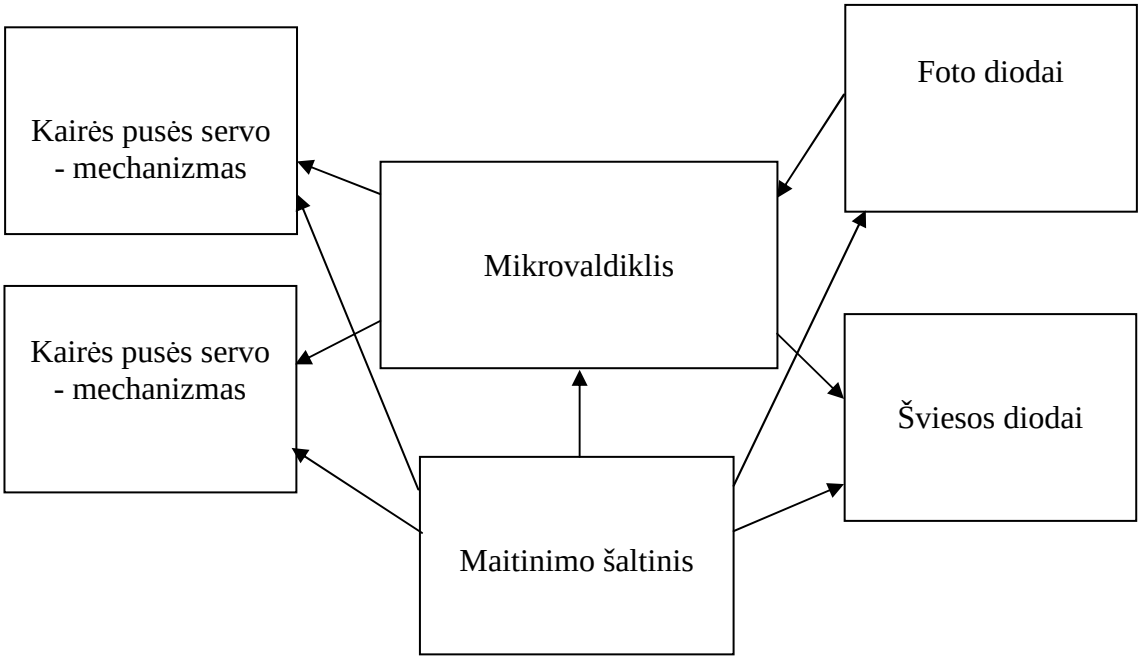


Pav. 11 WinAvr programos langas

Kodo įrašymui į mikrovaldklių buvo naudojamas USB ASP savadarbis programuotojas. Programuotojo ypatumas – palaikoma USB sąsaja, tiesioginis programavimas iš „Programmers Notepad“ programos lango ir „In Circuit Programming“, leidžianti programuoti pajungtą mikrovaldiklį.

Dėl realios užduoties testavimo trūkumo – lėto žolės augimo – dengiamo ploto žymėjimui buvo parinktas baltas popieriaus lapas. Kliūčių ir padengto ploto žymėjimui naudojama juoda spalva. Pagal pasirinktą modeliuojamą plotą buvo parinkti ir optiniai davikliai, kurie duotu atveju užtikrina pakankamą tikslumą ir funkcinį realių žoliapjovės modelių atitikimą.

Sukurto roboto blokinė schema parodyta [Pav 12]:



Pav. 12 Roboto blokinė schema

8.3 Sukurto roboto NPD algoritmo kūrimas

Pirmas etapas kuriant robotą, skirtą NPD algoritmo bandymui, buvo tarpinės sąsajos sukūrimas. Tarpinės sąsajos paskirtis – atskirti roboto modulių valdymą nuo NPD algoritmo. Tam tikslui buvo sukurtos šios paprogramės:

- void F(void) – Forward (pirmyn);
- void B(void) – Backward (atgal);
- void L(void) – Left (kairė);
- void R(void) – Right (dešinė);
- void S(void) – Stop (stop).

Paminėtos paprogramės nustato tolesnio judėjimo vektoriaus kryptį.

Nustatyto judėjimo vykdymui buvo sukurtos šios paprogramės:

- void go1(void) – vykdyti1;
- void go2(void) – vykdyti2.

Šios paprogramės nusako judėjimo vektoriaus ilgį. Judėjimo vektoriaus paprogramės nustato varančiųjų variklių valdiklių valdymo signalus, vektoriaus ilgio – komandos veikimo laiką. Kaip pavyzdį galima pateikti komandų seką, skirtą roboto judėjimui pirmyn. Komandų seka atrodytų taip :

```
F();
Go1();
```

Komada F(); nustato abiejų variklių sukimosi greitį judėjimui pirmyn, o komanda Go1(); leidžia varikliams suktis nustatytą laiką. Tokiu būdu robotas pajuda į priekį.

Paprogramių kūrimo metu buvo susidurta su tokiomis problemomis: panaudoti servo-mechanizmai nėra preciziški, jų naudojami valdikliai yra labai jautrūs trukdžiams ir valdomo signalo iškraipymui. Norint pasiekti tinkamų rezultatų, reikėjo atsisakyti WinAvr servo-mechanizmų valdymo bibliotekos ir kurti savo. Tiksliam signalui užtikrinti buvo naudotas vidinis 16 bitų taimeris. Panaudotas sprendimas padidino servo - mechanizmų valdymo tikslumą.

Kita problema buvo roboto valdymo paprogramių sukalibravimas. Tam tikslui prie roboto buvo pritvirtintas žymeklis. Robotas buvo užprogramuotas ciklišk ai vykdyti valdymo komandų sekas. Žymeklis žymėjo judėjimo trajektoriją. Bandymo būdu buvo sukalibruoti roboto servo - mechanizmų signalo lygiai kiekvienai judėjimo komandai. Taip pat bandymo būdu buvo parinkti optimalūs komandų veikimo laikai.

Judėjimo posistemės kalibravimui buvo naudojamas toks algoritmas:

```
While (1)
        {
                F();
                Go1();
        }
```

Algoritmas užtikrino pastovų roboto judėjimą į priekį, naudojant maksimalias judėjimo variklių apsukas. Robotui judant su nuokrypiu į dešinę pusę, buvo mažinami kairiojo variklio apsisukimai. Analogiškai robotui judant su nuokrypiu į kairę, buvo mažinami dešiniojo variklio apsisukimai.

Pasiekti rezultatai parodė pakankamą pasirinktos kinematinės posistemės veiksmingumą. Norint pasiekti puikių rezultatų, reikėtų naudoti kinematinę posistemę su atgaliniu ryšiu. Tokiu būdu galėtume tikėtis tikslesnio roboto valdymo.

Jutiklinei posistemei yra panaudoti optiniai davikliai. Tam tikslui roboto apačioje yra įmontuoti foto ir šviesos diodai. Foto diodų signalai yra pajungti į mikrovaldiklio įėjimo portus. Pirmutiniame algoritmo variante foto daviklių signalai buvo nuskaitomi pastoviai. Dėl daviklių jautrumo matomai šviesai reikėjo įvesti foto diodų signalo kalibraciją, kuri pašalintų išorinės šviesos poveikį. Tam tikslui prieš pradedant matuoti foto daviklių signalą yra išjungiami šviesos diodai, pamatuojamas foto diodų signalo lygis, jis išsaugomas, tada įjungiami šviesos diodai ir vėl pamatuojamas foto diodų signalo lygis. Jeigu signalų skirtumas telpa į nustatytas paklaidos ribas ir pradinis daviklio apšvietimas yra mažesnis už nustatytąjį, spėjama, kad robotas yra virš nepadengto ploto – balto lapo. Priešingu atveju – robotas yra virš padengto/ribinio ploto.

Norint pasiekti tinkamą posistemės veikimą, reikėjo parinkti foto/šviesos diodų tarpusavio išdėstymą, foto diodų roboto korpuso apačioje išdėstymą, parinkti tinkamą valdymo signalo lygį. Dar daugiau reikėjo padirbėti prie jutiklinės posistemės kodo – reikalingas tinkamas klaidingų duomenų atpažinimas, skirtingo foto diodų jautrumo lygio kalibravimas, apsauga nuo išorinio apšvietimo.

Pradinis jutiklinės posistemės kalibravimas atliktas pamatavus fotodaviklių signalo lygį prie žemiau pateiktų sąlygų:

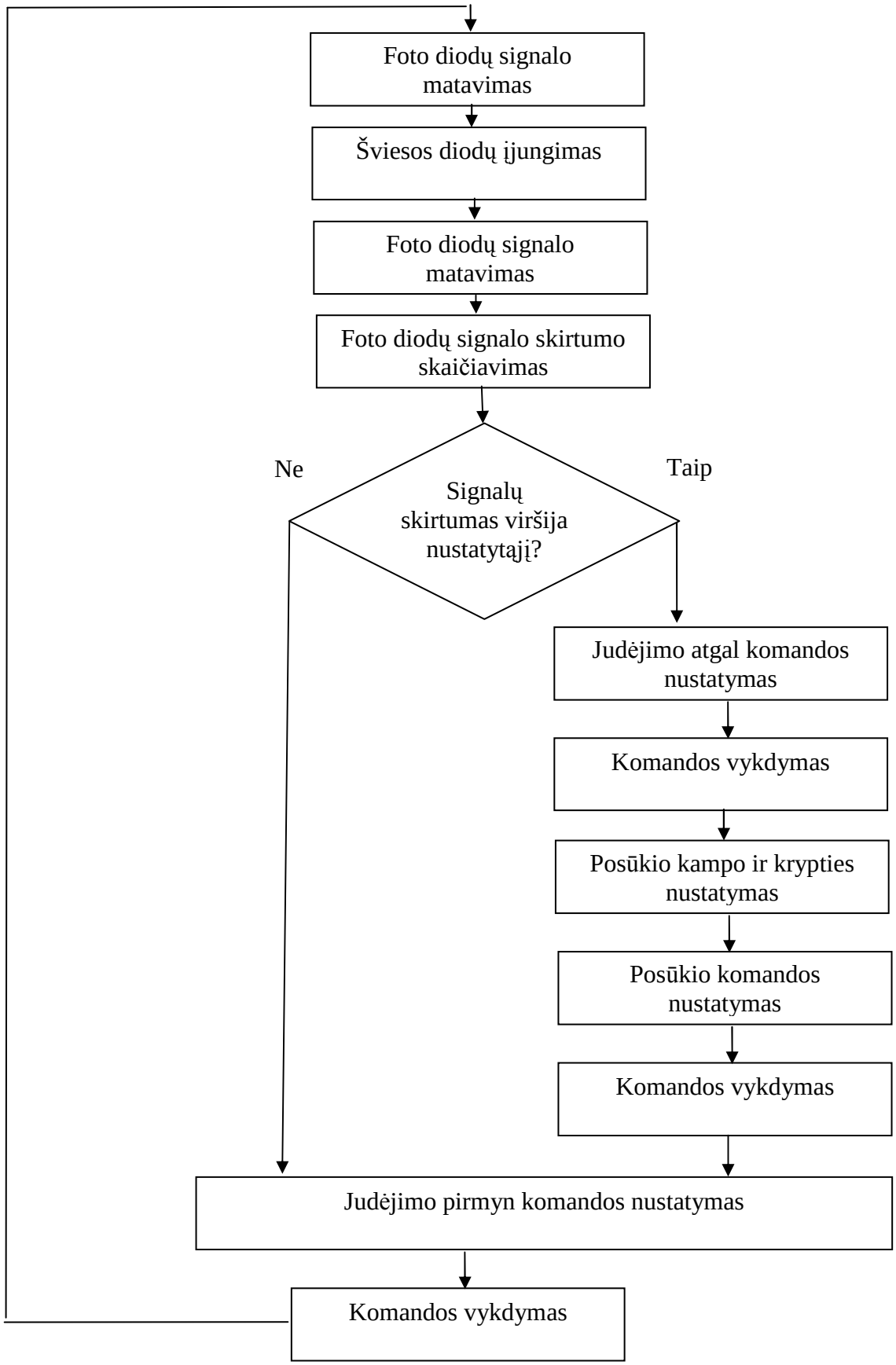
1. virš balto popieriaus lapo su išjungtu šviesos diodu;
2. virš juodo popieriaus lapo su išjungtu šviesos diodu;
3. virš balto popieriaus lapo su įjungtu šviesos diodu;
4. virš juodo popieriaus lapo su įjungtu šviesos diodu.

Kaip ir buvo tikėtasi, signalo lygis 1, 2 ir 4 atveju skyrėsi nežymiai ir vidutiniškai buvo lygus 50, matavimui naudojant mikrovaldiklio vidinį analoginį - skaitmeninį keitiklį (ASK). Atveju Nr. 3, ASK signalas vidutiniškai sudarė 220. Valdymui naudojant signalo skirtumą tarp 1 ir 3, ir tarp 2

ir 4 atvejų, parinktas pradinis ribinis foto diodo signalo pokytis lygus 160. Jutiklinės posistemės bandymo metu su pasirinktu foto diodo signalo pokyčiu pastebėtas nepakankamas posistemės jautrumas ir neatsparumas išorės apšvietimui. Bandymų būdu foto diodų signalo skirtumas buvo sumažintas iki 140, ir tai užtikrino pakankamą posistemės jautrumą bei atsparumą trukdžiams.

Nors pasiekti rezultai parodė jutiklinės sistemos veiksmingumą, bet, norint užtikrinti patikimumą ir kartotinumą, reikėtų naudoti tokius daviklius, kurie nebūtų jautrūs išoriniam apšvietimui, arba apsiriboti jutimo davikliais.

Turint sukurtas paprogrames, sukalibruotą jutiklinę ir kinematinę posistemę, NPD algoritmo sukūrimas buvo trivialus. Panaudotas daviklių išdėstymas leisdavo nuskaityti atsimušimo į kliūtį kampą. Kinematinė posistemė, savo ruožtu, leisdavo pasukti robotą reikiamu kampu ir judėti tiesiai. Grafiškai algoritmas atrodytų taip [Pav 13]:



Pav 13. Roboto algoritmo schema

8.4 Pasiekti rezultatai

Sukurtas robotas leido gyvai testuoti „Bouncing“ algoritmą. Algoritmas parodė, kaip sprendžiamas realus NPD uždavinys, esant vidutinio/žemo tikslumo kinematinei ir jutiklinei posistemei. Taip pat buvo parodyta galimybė programiškai atlikti daviklių kalibravimą, klaidingų duomenų atmetimą.

Panašaus roboto sukūrimo savikaina neviršijo 100 lt. Darbo metu buvo sukurtas paprastas robotas, kurio pagaminimas nereikalauja daug įgūdžių, ir nereikia daug žinių perprasti jo veikimą. Dėl mažos kainos, sudėtingumo ir galimybės vykdyti sudėtingus algoritmus atskirai nuo roboto posistemių valdymo, robotas gali būti naudojamas kaip platforma įvairių algoritmų tyrimui.

9 Ateities tikslai

Sukurtas robotas parodė realų NPD algoritmo veikimą. Norint tiksliau nagrinėti algoritmų efektyvumą, reikėtų įvesti roboto judėjimo trajektorijos žymeklį. Tam tikslui gali būti panaudotas žymeklis, matomas tik ultravioleto šviesoje, arba video kamera sumontuota virš modeliuojamo ploto. Tokiu būdu galima tiksliau išnagrinėti algoritmą/-us, gauti statistinių duomenų.

Kitas aspektas – roboto programavimo patogumas. Dabar tam tikslui reikia prijungti robotą prie programuotojo. Norint programavimą padaryti patogesnį, reikėtų sukurti bevielio programavimo įrangą. Tam tikslui galima panaudoti infraraudonuosius spindulius arba radio bangas. Abiem atvejais yra gatavi imtuvo/siųstuvo moduliai. Taip pat bevielio ryšio sistema leistų kaupti statistinius duomenis programos vykdymo metu.

Norint paįvairinti tyrimo sritį, galima sukurti keletą robotų. Tokiu būdu galima modeliuoti kelių robotų NPD sprendimą. Toks papildymas atitinkamai pareikalautų ir robotų tarpusavio komunikacijos.

10 Išvados

Pademonstruotas NPD algoritmo veikimas realiame robote. Parodytas nesudėtingo NPD „Bouncing“ algoritmo veikimas esant mažo tikslumo jutiklinei ir kinematinei posistemei. Sukurta jutiklinė posistemė, galinti atrinkti klaidingus duomenis, atlikti jautrumo kalibravimą, priklausomai nuo išorinės aplinkos trukdžių. Atliktas darbas parodė sudėtingus roboto kūrimo aspektus ir galimus sprendimus. Taip pat sukurtas robotas gali būti naudojamas ir kaip platforma kitų algoritmų kūrimui/testavimui.

Atliktas NPD praktinio naudojimo – robotizuota žoliapjovė – parodė naudojamus sprendimus, jų silpnąsias ir stipriąsias puses. Rasti tinkami posistemių sprendimai, skirti robotozuotai žoliapjovei.

Pastebėtas praktinis NPD problemos aktualumas. Pradėjus rašyti darbą buvo žinoma tik keliolika robotizuotų žoliapjovių. Po dviejų metų jų atsirado kelios dešimtys. Taip pat atsirado pasiūla Lietuvoje.

11 Literatūros sąrašas

[Lav06] Steven M. LaValle. Planning algorithms, 2006

[Cho01] Howie Choset. Coverage for robotics – A survey of recent results, 2001

[GP06] G. Gutin and A. P. Punnen. The Travelling Salesman Problem and Its Variations. Springer, 2006

[MJ04] Manuel Mazo Jr., Karl Henrik Johansson. Robust area coverage using hybrid control, 2004

[CP97] H. Choset, P. Pignon. Coverage Path Planning: The Boustrophedon Cellular Decomposition, 1997

[Cho00] H. Choset. Coverage of known spaces : The Boustrophedon Cellular Decomposition, 2000

[AC01] E. Acar, H. Choset .Robust Sensor-based Coverage of Unstructured Environments, 2001

[BRH00] Z. Butler, A. Rizzi, R. Hollis. Cooperative Coverage of Rectilinear Environments, 2000

[WCM00] S. Wong, G. Coghilland, B. Mac Donald. Landmark-based world model for autonomous vacuuming robots, 2000

[Cho02] Howie Choset. Auto-Body Painting, 2002

[CK99] Kok Seng CHONG Lindsay, KLEEMAN. Mobile Robot Map Building from an Advanced Sonar Array and Accurate Odometry, 1999