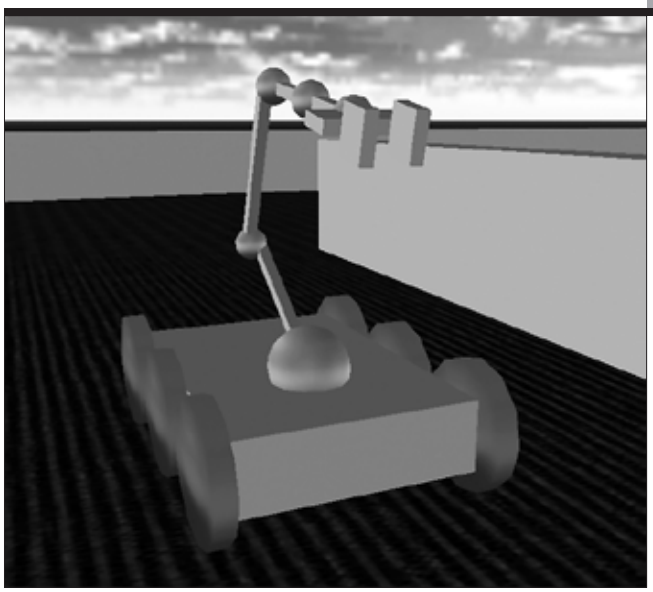


Autonoma robota modelis

Dr. sc. ing. **Agris Ņikitenko**

1. attēls. Robota imitācijas modeļa grafisks attēls

Raksts veltīts LR Aizsardzības ministrijas finansēta projekta AM 2007/52 galvenajiem rezultātiem un risināmajām problēmām, kas saistītas ar autonomu robotu izstrādi un vadību. **Raksta ievadā dots problēmas apraksts, bet atlikušajā daļā aprakstīts izmantotais risinājums autonoma robota vadībai. Rakstā īsumā aprakstītas galvenās izmantotās metodes un to pielietojuma īpatnības. Nobeigumā ir apskatīti turpmāko pētījumu virzieni un ar tiem saistītie izaicinājumi.**

Atslēgvārdi: Mākslīgais intelekts, Robotika, Autonomas sistēmas

Ievads

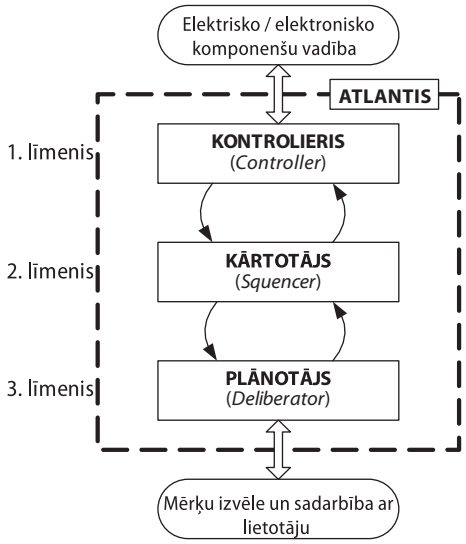
Autonomija kā termins ir salīdzinoši ietilpīgs un vispārīgs, kas var tikt attiecināts uz dažāda veida sistēmām, gan cilvēka radītām, gan dabiskām. Šī raksta ietvaros autonomija tiek attiecināta tikai uz cilvēka radītām sistēmām – robotiem.

Attīstoties robotikas jomai, arvien vairāk tiek paplašinātas funkcijas, kas tiek uzticētas robotiem, bez cilvēka tiešas uzraudzības un palīdzības. Autonoma robota galvenā īpašība ir spēja veikt uzticētos uzdevumus bez ārējas palīdzības neatkarīgi no palīdzības sniedzēja – cilvēka vai citas mākslīgas sistēmas. Īpaši militārajā jomā autonomie roboti mūsdienās gūst plašu pielietojumu, jo ļauj mazināt draudus cilvēkiem, saglabājot augstu uzdevumu izpildes efektivitāti. Lai arī notiek aktīvs darbs autonomo robotu jomā, pašlaik nevar apgalvot, ka mūsdienu tehnoloģija ļauj izstrādāt pilnīgi autonomus robotus, kas patstāvīgi spēj veikt uzdevumus, atjaunot savus enerģijas krājumus, diagnosticēt un remontēt savu apakšsistēmu bojājumus u. tml. Tādēļ parasti ir runa par autonomiju konkrēta uzdevuma vai uzdevumu kategorijas robežās, kura paredz zināmu mērķu un to sasniegšanas alternatīvu izvēles brīvību [Riegler_2008, Willem_2005].

Autonomu robotu galvenā īpašība ir mobilitāte, kas ļauj robotam pārvietoties, lai veiktu sev uzticētos uzdevumus. Pārējās robotu īpašības un funkcijas tiek realizētas, galvenokārt pateicoties mobilitātei. Projekts tika veltīts sauszemes robotu pētīšanai, kuriem piemītošā mobilitāte ļauj veikt šādus galvenos uzdevumus:

- objektu vai teritorijas apsardze un patrulēšana teritorijā;
- objekta vai teritorijas izpēte;
- telpu apsekošana;
- teritorijas kartes un telpu plāna sastādīšana;
- kustīgu objektu novērošana un izsekošana;
- specifisku objektu meklēšana konkrētā teritorijā;

Lai arī kāds uzdevums no minētajiem tiek uzticēts robotam, galvenā problēma ir robota navigācija un savas atrašanās vietas precīza noteikšana, izmantojot uzdevuma ietvaros pastāvošu globālu koordinašu sistēmu, t. i., robota atrašanās vieta un pozīcija tiek noteikta attiecībā pret dabā novērojamu objektu vai iedomātu punktu.



2. attēls. ATLANTIS trīs līmeņu arhitektūras shēma

Lai risinātu minēto problēmu, projekta ietvaros tika izstrādāts autonoma robota imitācijas modelis, izmantojot 3D grafikas un dinamikas modelēšanas iespējas, ko piedāvā robotu programmēšanas rīks *MS Robotics Studio 1.5* [Robotics_2008].

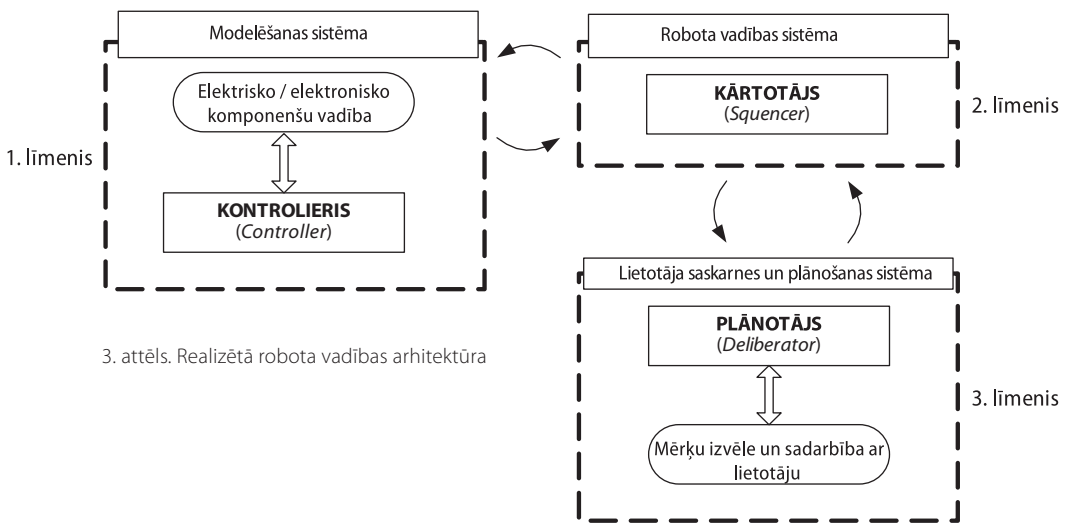
Izstrādātā robota imitācijas modelis redzam 1. attēlā.

Robota arhitektūra

Izstrādātā robota imitācijas modeļa (turpmāk tekstā *robots*) arhitektūras pamatā ir labi pazīstamā trīs līmeņu arhitektūra ATLANTIS, kas paredz visus robota vadības uzdevumus sadalīt trijās kategorijās atbilstoši uzdevumu sarežģītībai un paredzamajam skaitļošanas ilgumam [Gatt_1992]. Šādi tiek nodrošināta asinhrona un heterogēna robota vadība. Arhitektūra redzama shēmā 2. attēlā.

Mūsdienās 1. līmeņa funkcijas tiek nodrošinātas, izmantojot dzinēju kontrolierus, kas ļauj izpildīt tipveida darbības, piemēram, iedarbināt motoru uz noteiktu laiku ar konkrētu griešanās ātrumu. Ņemot vērā to, ka projekta ietvaros tika izmantots imitēts robots, t. i., netika izmantoti reāli motori un atbilstoši kontrolieri, lielākās pūles tika veltītas 2. un 3. līmeņa funkcijām, kā arī sadarbībai ar lietotāju. Atbilstoši ATLANTIS arhitektūrai 2. līmenis nodrošina darbību secību izpildi, kurā katra darbība nav tieši saistīta ar izpildmehānismu vadību, piemēram, braukt uz priekšu vienu metru, pagriezties pa labi par 45 grādiem u. tml. [Gatt_1992] Robota 2. līmeņa vadības modulis satur sarakstu ar darbībām, ko tas spēj izpildīt. Tieši tāds pats saraksts ir 3. līmeņa vadības modulī, kurš veic darbību plānošanu. Darbību plāns tiek nosūtīts Kārtotājam uz izpildi. Kārtotājs ir atbildīgs par visa plāna izpildi. Ja kāda no darbībām izpildes laikā cieš neveiksmi, tad Kārtotājs, nosūta atbilstošu ziņojumu Plānotājam, kuram jāveic atkārtota plānošana. Papildus minētajam, Kārtotājs ir atbildīgs par navigācijas informācijas izgūšanu no sensoru datiem, kā arī par šīs informācijas regulāru nosūtīšanu Plānotājam.

ATLANTIS arhitektūra paredz ka visi trīs līmeņi ir iebūvēti robotā. Programmatūras izstrādes ērtības labad, projekta ietvaros 2. un 3. līmeņa vadības moduļi tika atdalīti un izstrādāti kā atsevišķas programmatūras sistēmas. Programmatūras arhitektūras modelis un funkciju sadalījums starp līmeņiem attēlots 3. attēlā redzamajā shēmā.



3. attēls. Realizētā robota vadības arhitektūra

Ar raustītu līniju 3. attēlā iezīmētas atsevišķas programmatūras sistēmas, t. i., kopā 3. Kā minēts iepriekš, projekta ietvaros izstrādātas tika 2. un 3. līmenim atbilstošās sistēmas. Funkciju sadalījums starp realizētajām programmatūras sistēmām ir šāds:

1. Robota vadības sistēma:
   a. Darbību izpilde saskaņā ar plānu;
   b. Darbību izpildes rezultātu novērošana, novērtēšana un izpildes rezultātu nosūtīšana Plānotājam;
   c. Robota atrašanās vietas un pozīcijas noteikšana – pašlokalizācija [Martinez_2003];
2. Lietotāja saskarnes un plānošanas sistēma:
   a. Kārtējā plāna sastādīšana;
   b. Kartes sastādīšana;
   c. Mērķu alternatīvu izvēle – mērķus izvēlas lietotājs no šādām alternatīvām: iet uz konkrētu vietu, norādot to kartē, patrulēt teritoriju, izpētīt teritoriju. Teritoriju lietotājs norāda, kartē atzīmējot interesējošās teritorijas robežpunktus.

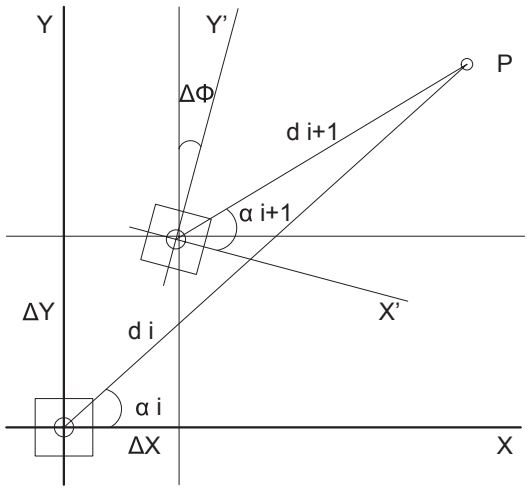
Matemātiski sarežģītākās funkcijas, bez robota kustības matemātiskā modeļa, ir pašlokalizācija un darbību plāna sastādīšana. Nākošajā nodaļā īsumā aplūkota šo problēmu risinājumi projekta ietvaros.

Pašlokalizācija

Pašlokalizācija ir svarīgākā risināmā problēma, bez kuras maršruta plānošana nav iespējama. Projekta ietvaros tiek izmantots robots, kura kustība ir salīdzināma ar tanku, t. i., kustības virziens tiek koriģēts, samazinot vai palielinot vienā robota pusē esošo riteņu kustības ātrumu. Lai arī šāda robota konstrukcija un vadība ir salīdzinoši vienkārša, pašlokalizācija ir izaicinājums, jo pretstatā Akermana vai diferencētai piedziņai [Borenstein_1996], robotam pagriežoties, tā riteņi slīd. Slīdēšana ir atkarīga no virsmas saķeres ar riteņiem. Lai arī ir mēģinājumi precīzi aprakstīt konkrēta robota riteņu (vai ķēžu) slīdēšanas īpašības [Martinez_2005], tas arvien ir aktīvu pētījumu priekšmets. Tādēļ ir apgrūtināta precīza matemātiskā modeļa izstrāde.

Projekta ietvaros tika izmantota Martineza [Martinez_2003] piedāvātā metode, kas par pamatu izmanto lāzera attāluma mērītājus robota pārvietojuma noteikšanai, apstrādājot divus savstarpēji sekojošus skanējumu datus. Piedāvātā metode ir pielietojama tikai divu dimensiju telpām. Projekta

4. attēls. Pašlokalizācija, izmantojot 2D lāzera skeneri



ietvaros tika izmantota tieši šāda tipa telpa, jo robots pārvietojās pa līdzenu virsmu.

Metodes būtība atklāta 4. attēlā redzamajā shēmā [Martinez_2003].

Tiek veikts viens lāzera skenējums, kurā tiek pieņemts, ka robots atrodas koordināšu sākuma punktā ar rotācijas leņķi 0. Ir jāuzsver, ka koordināšu sākuma punkts ir lokālajās koordinātēs, t. i., robotam pārvietojoties, pārvietojas arī šis sākuma punkts (4. attēlā – no XY plaknes uz plakni X'Y'). Skanējumā tiek atklāts viens punkts P (šeit vienkāršības labad tiek aplūkots tikai viens), kas atrodas attālumā di no koordināšu sākuma punkta un virzienā α i attiecībā pret X asi. Pēc salīdzinoši īsa laika (10–100ms) tiek veikts nākošais skanējums, kurā tiek noteikts punkts P, bet attālumā di+1 un virzienā α i+1 lokālajās koordinātēs X'Y'.

Ja ir zināms robota pārvietojums T = (ΔX, ΔY,ΔΦ), tad, izmantojot vienkāršus trigonometrijas aprēķinus, var aprēķināt robota sākuma pozīciju pirmā skenējuma laikā. Tas nozīmē, ka α i+1 = f(T) + α i un di+1 = g(T) + di, kur funkcijas f un g ir attiecīgi trigonometriski pārveidojumi, kas sīkāk apskatīti [Martinez_2003, Martinez_2005].

Problēma ir tāda, ka pārvietojums T nav zināms, tas ir jāaprēķina. Ja tiek aplūkots nejauši izvēlēts pārvietojums T_k, tad pēc funkciju f un g izmantošanas, rodas kļūdas:

$$\xi\alpha = \alpha_{i+1} - f(T) - \alpha_i \text{ un } \xi d = d_{i+1} - g(T) - d_i$$

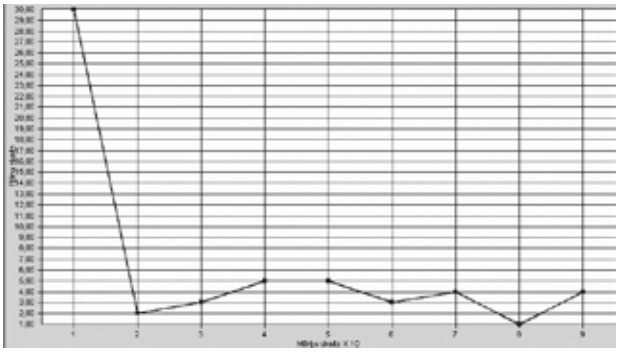
Kad atrisināts optimizācijas uzdevums, kurā minimizē minētās kļūdas, tad tiek iegūts pārvietojums T_k ar pieņemamu kļūdas lielumu. Ilgākā laika posmā pie zināma kustības sākuma punkta

globālajās koordinātēs, integrējot T_k pārvietojumus, tiek iegūts kopējais pārvietojums, no kura tiek aprēķināta robota atrašanās vieta un orientācija globālajās koordinātēs. Martinezs piedāvā izmantot ģenētisko algoritmu kā optimizācijas tehniku, kas ļauj atrast pieņemamu atrisinājumu T_k. Metodes vājā puse ir kļūdas uzkrāšanās, kas SLAM (no angļu val. *Simultaneous localization and mapping*) metožu kontekstā parasti tiek novērsta, izmantojot kādu no daļiņu filtriem (*particle filter*), piemēram, Rao-Blackwellized filtru [Eliazar_2004, Dong_2007]. Projekta ietvaros daļiņu filtri netika piemēroti, jo šis uzdevums tiek risināts turpmāko pētījumu ietvaros (skat. zemāk).

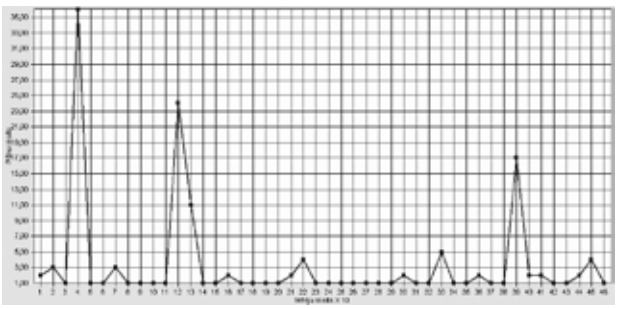
Pašlokalizācijas rezultātā iegūtie pārvietojuma dati tiek izmantoti, lai noteiktu robota atrašanās vietu un plānotu turpmākās darbības, kā arī lai noteiktu, vai izpildāmās darbības ir izpildītas veiksmīgi.

Darbību plānošana

Uzsākot projektu, robota vadības sistēmas koncepcija tika balstīta uz iepriekšējos pētījumos izstrādātas hibrīdas intelektuālas sistēmas kodola arhitektūras [Nikitenko_2007, Nikitenko_REM_2007], kas nodrošināja sistēmas salīdzinoši strauju apmācību un spēju pielāgoties. Tomēr rūpīgāki pētījumi atklāja, ka ilgākā darbībā sistēma iziet no stabila stāvokļa, t. i., sistēma nonāk tā sauktajā pārapmācītā stāvoklī. Šis stāvoklis ir raksturīgs ar to, ka apmācības rezultātā tiek iegūti pārāk specifiski likumi, kas kļūst nederīgi iepriekš nepieredzētā situācijā. Šī problēma skāra tikai darbību plānotāja un apmācības mehānismu. Pārapmācības stāvoklis iestājas tad, kad nav mehānismu, kas precīzi nosaka, kad sistēma ir adaptējusies pietiekami labi, lai adaptēšanās mehānismi vairs netiktu darbināti. Pārraudzītas apmācības gadījumā [Russell_2003], kad ir zināmi pareizie risinājumi, un tiek saņemti sistēmas sniegtiem risinājumiem, par precīzu indikatoru ir atbilstības mērs starp pareizajiem risinājumiem un sistēmas risinājumiem. Vienkāršākā gadījumā tā ir attiecība starp kopējo risinājumu skaitu un sistēmas sniegtajiem pareizajiem risinājumiem: P = S / N, kur P apmācības novērtējums, S pareizo sistēmas risinājumu skaits, N – kopējais risinājumu skaits. Autonomu robotu vadības uzdevumos parasti ir grūti noteikt pareizos risinājumus, jo vide ir sarežģīta, dinamiska un grūti prognozējama [Nikitenko_2006]. Tas nozīmē, ka nav formāla



5. attēls. Pārapmācības problēma (A – salīdzinoši ātra apmācība)



6. attēls. Pārapmācības problēma (B – pēc ilgāka laika sistēma nonāk nestabilā stāvoklī, kas prasa papildus mēģinājumus)

mehānisma, kas ļauj *apriori* noteikt, kādas darbību secības ir pareizas vai nepareizas iespējamās nākotnes situācijās.

Uzskatāmi minētās problēmas novērojumu rezultāti ir redzama šādos grafikos (5., 6. attēls), kur uz x ass ir norādīts sasniegto mērķu skaits, bet uz y ass mēģinājumu skaits (katra mēģinājuma laikā tiek izpildīts konkrēts plāns, lai robots pārvietotos uz norādīto pozīciju).

Diemžēl novērotā pārapmācības problēma netika atrisināta projekta ietvaros, jo projektam ir ierobežots laiks.

Lai atrisinātu šo problēmu, tika izvēlēts cits plānošanas algoritms, kas nodrošina zināmas adaptēšanās spējas vides īpatnībām – šķēršļiem. Projektā tika aplūkotas divas alternatīvas: ģenētiskie algoritmi un RRT plānotājs (*Rapidly-Exploring Random Trees*) [Bruce_2003]. Abi algoritmi nodrošina salīdzinoši ātru darbu, kas ir īpaši nepieciešams robotu pielietojumos, jo tie darbojas reāla laika apstākļos. Veicot literatūras izpēti, tika izvēlēta RRT pieeja, kas nodrošina labāku kontroli pār plāna ģenerēšanas procesu [Bruce_2003, Doyle_1995]. Līdzīgi ģenētiskajiem algoritmiem RRT plānotājs nenodrošina optimālu plānu, bet nodrošina pieņemamu plānu, kas ved uz mērķi, ja tas ir sasniedzams pie noteiktajiem ierobežojumiem, piemēram, maksimālā maršruta garuma. Plānotājs sava darba laikā ģenerē maršrutu, kas sastāv no soļiem un pieturas

punktiem, kas savieno divus soļus. Plānotājs realizē šādus galvenos soļus:

1. Izvēlas galamērķim vistuvāko pieturas punktu. Attālums tiek noteikts, aprēķinot Eiklīda attālumu līdz galamērķim.
2. Izvēlas kustības mērķi – izvēle notiek ar norādītu varbūtību, t. i., ar varbūtību p tiek izvēlēts lietotāja norādītais mērķis, bet ar varbūtību 1-p nejaušs mērķis.
3. Tiek sperts fiksēts solis izvēlētā mērķa virzienā.

Šādi tiek nodrošināta salīdzinoši ātra algoritma darbība. Varbūtīgs mērķa izvēles mehānisms ļauj izvairīties no lokāli optimāliem risinājumiem. Sīkāk algoritma darbība un izmantošana robotu navigācijai aplūkota avotā [Bruce_2003].

Kombinējot pašlokalizāciju un darbību plānošanu, ir izveidoti algoritmi teritorijas izpētei un patrulēšanai. Tie aplūkoti nākošajā nodaļā.

Patrulēšana un teritorijas izpēte

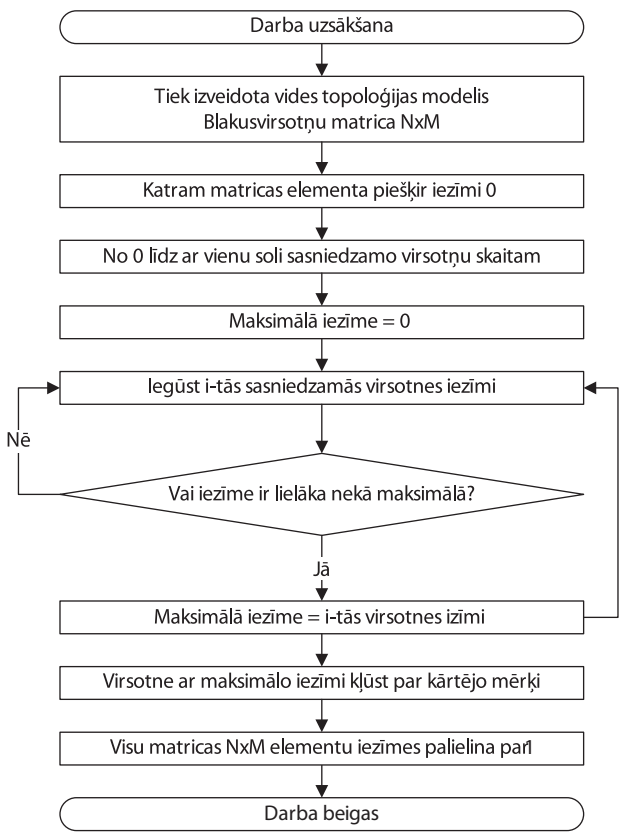
Lietotāja saskarnes un plānošanas sistēma (skat. 3. attēlu) nodrošina lietotājam iespēju norādīt augsta līmeņa uzdevumus, kas robotam jāizpilda. Šim nolūkam tiek izmantota ģeogrāfiskās informācijas sistēma *Map Windows GIS* [MapWindows GIS_2008], kas lietotājam nodrošina iespēju ar datora peli norādīt punktus kartē, kā arī aplūkot robota atrašanās vietu un šķēršļus.

Projekta ietvaros tika realizēti algoritmi šādu uzdevumu veikšanai:

- Pārvietošanās uz konkrētu pozīciju, kuru lietotājs norāda kartē. Pēc tam tiek iedarbināts plānotājs, kas izveido darbības plānu. Plāns tiek nosūtīts Kārtotājam uz izpildi.
- Teritorijas izpēte – lietotājam kartē jānorāda vismaz 3 kontrolpunkti, kas nosaka izpētāmās teritorijas robežas. Katrs no punktiem tiek apmeklēts vienu reizi, par kārtējo kontrolpunktu izvēloties to, kas atrodas vistālāk no robota atrašanās vietas – kontrolpunkta, kuru robots sasniedzis. Šādi tiek palielināts kopējais informācijas daudzums, ko robots iegūst, pārvietojoties starp kontrolpunktiem.
- Patrulēšana teritorijā – lietotājam kartē jānorāda vismaz 3 kontrolpunkti, starp kuriem robotam jāveic regulāra patrulēšana. Pretstatā teritorijas izpētei, patrulēšanas uzdevumā robotam regulāri jāapmeklē katrs no lietotāja norādītajiem kontrolpunktiem, ievērojot, ka pārvietošanās laikā, lietotājs var mainīt kontrolpunktus, tos pievienojot vai izslēdzot no kontrolpunktu saraksta. Patrulēšanas uzdevums matemātiski ir aprakstāms kā ceļojošā pārdevēja uzdevums [Anisi_2007, Fakcharoenphol_2007], kas aprēķinu ziņā ļoti laikietilpīgs, ja ir daudz kontrolpunktu. Ir jāuzsver, ka patrulēšanas uzdevuma būtība ir samazināt laiku starp katra kontrolpunkta diviem sekojošiem apmeklējumiem. Mūsdienās tiek izmantotas dažādas tehnikas šī uzdevuma risināšanai, kas, galvenokārt, to aplūko kā optimizācijas uzdevumu un risina, izmantojot atbilstošus matemātiskos paņēmienus [Anisi_2007, Fakcharoenphol_2007]. Parasti šādi risinājumi ir salīdzinoši laikietilpīgi, lai arī nodrošina risinājumu, kas ir tuvs optimālajam. Kā minēts iepriekš, roboti darbojas reāla laika apstākļos ar ierobežotiem skaitļošanas resursiem. Tas nozīmē, ka tiem var nebūt pieejami nepieciešamie resursi šādu skaitļošanas uzdevumu risināšanai.

Tādēļ projekta ietvaros tika izstrādāta ievērojami vienkāršāka pieeja, kas ļauj ar nelieliem skaitļošanas resursiem reāla laika apstākļos veikt patrulēšanu. Algoritma blokshēma parādīta 7. attēlā.

7. attēls. Patrulēšanas algoritma blokshēma



Izveidotais patrulēšanas algoritms nenodrošina optimālu risinājumu, tomēr, ļauj ātri izvēlēties kārtējo kontrolpunktu, uz kuru doties. Algoritms tiek darbināts, katru reizi, kad ir sasniegts kārtējais patrulēšanas kontrolpunkts un ir nepieciešams izvēlēties nākošo. Katru reizi tiek izveidots jauns vides topoloģiskais modelis – grafs, kas izteikts, izmantojot kontrolpunktu blakusvirsotņu matricu. Tādēļ vienmēr tiek ņemtas vērā pēdējās izmaiņas kontrolpunktu sarakstā.

Turpmākie pētījumi

Projekta ietvaros tika izveidota programmatūras bāze turpmākajiem pētījumiem autonomu sistēmu jomā, nodrošinot pašus galvenos mehānismus – paš-lokalizāciju un darbību plānošanu, kā arī papildu mehānismus augsta līmeņa uzdevumu veikšanai, t. i., teritorijas izpēti un patrulēšanu. Turpmākajos pētījumos lielākās pūles tiks veltītas šādu problēmu risināšanai:

– Robota prototipa izstrāde. Lai veiktu kvalitatīvus turpmākos pētījumus, ir nepieciešams robota prototips, kas kalpos par eksperimentālo bāzi vadības sistēmu izstrādei un to darbības pārbaudei.
– Robota dinamikas matemātiskā modeļa izstrāde. Projekta ietvaros tika izmantots vienkāršots matemātiskais modelis, kas neļauj to izmantot reāla robota vadībai. Kā minēts iepriekš, projektā izmantotās pašlokalizācijas metodes trūkums ir kļūda, kas uzkrājas ilgākā darbības laika posmā. Lai pielietotu precīzākas SLAM metodes, ir nepieciešams robota dinamikas matemātiskais modelis, kura sniegtie pārvietojuma dati tiek kombinēti ar sensoru datiem. Pašlaik notiek darbs pie šāda modeļa izstrādes, kas tiks balstīts uz Martineza piedāvātā modeļa [Martinez_2005], to atbilstoši pielāgojot konkrēta robota vajadzībām.
– Projekta ietvaros izmantotā maršruta plānošanas metode RRT ir labi piemērota diskrētu darbību plānošanai. Tomēr robota kustības efektivitātes palielināšanai ir nepieciešams realizēt nepārtraukta maršruta plānošanu. RRT plānotājs šādiem mērķiem ir vāji piemērots [Bruce_2003]. Tādēļ turpmākajos pētījumos ir jāizstrādā nepārtraukta maršruta plānotājs.
– Jāturpina izstrādāt un pilnveidot integrēta programmatūras sistēma, kas ļauj efektīvi vadīt robotu, kā arī sadarboties ar lietotāju, ļaujot norādīt augsta līmeņa uzdevumus. Šī uzdevuma veiksmīgai realizācijai ir jāturpina sadarbība ar projekta pasūtītāju, kas ļaus precīzi definēt robota veicamos uzdevumus un atbilstoši pielāgot vadības programmatūru.

LITERATŪRAS SARAKSTS

[Anisi_2007] D. A. Anisi. *Survey of patrolling algorithms for surveillance UGV*, Scientific report, 2007. Swedish Defence Research Agency.

[Borenstein_1996] J. Borenstein. Where am I? In: *Systems and Methods for Mobile Robot Positioning.* A. K. Peters, Ltd., Wellesley, MA, 1996, 282 p.

[Bruce_2003] J. Bruce. Real-Time Randomized Path Planning for Robot Navigation. 2003, Springer Lecture Notes in *Computer Science* Volume 2752/2003, pp. 288–295.

[Dong_2007] J. F. *Dong An Efficient Rao-Blackwellized Genetic Algorithmic Filter for SLAM.* IEEE International conference on Robotics and Automation, 2007, pp. 2427–2432.

[Doyle_1995] A. B. Doyle. *Algorithms and Computitional Techinques for Robot Path Planning.* PhD thesis, University of Wales, UK.

[Eliazar_2004] A. I. Eliazar, R. Parr. DP-SLAM 2.0, Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on Robotics and Automation, 2004, vol. 2, pp. 1314–1320.

[Fakcharoenphol_2007] J. Fakcharoenphol. The k-Travelling Repairmen Problem. *ACM Transactions on Algorithms*, 2007, Vol. 3, Num 4, Article 40.

[Gatt_1992] E. Gatt. *Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots*. Proceedings, 10th National Conference on Artificial Intelligence, AAAI-92;

[MapWindows GIS_2008] http://www.mapwindow.org/ Citēts: 2008. gada 30. novembrī

[Martinez_2003] J. L. Martínez. *Mobile robot self-localization by matching successive laser scans via genetic algorithms*. Proceedings of the 5th IFAC International Symposium on Intelligent Components and Instruments for Control Applications, Aveiro, Portugal, 2003.

[Martinez_2005] J. L. Martínez. Approximating Kinematics for Tracked Mobile Robots. *International Journal of Robotics Research*, 2005, Volume 24, Issue 10, pp. 867–878.

[Nikitenko_2006] A. Nikitenko. *Autonomous intelligent agent control in complex environment.,* Barcelona, Spain, EMSS 2006 Conference Proceedings, 2006, pp. 251–260.

[Nikitenko_2007] A. Nikitenko. *Knowledge-based robot control*. Varna, Bulgaria, KDS 2007 Conference proceedings, 2007, vol. 2, pp. 487–500.

[Nikitenko_REM_2007] A. Nikitenko. *Autonomous robot navigation using knowledge-based control unit*. Tallinn, Estonia, REM 2007 Workshop proceedings, 2007, pp. 93–98.

[Riegler_2008] A. Riegler. The paradox of autonomy: The interaction between humans and autonomous cognitive artifacts. In: Dodig-Crnkovic, G. & Susan Stuart, S. (eds.) *Computing, philosophy, and cognitive science. The nexus and the liminal.* Cambridge Scholars Publishing: Cambridge, 2008, pp. 292–301.

[Robotics_2008] http://msdn.microsoft.com/lv-lv/library/bb648760(en-us).aspx Citēts: 2008. gada 30. novembrī

[Russell_2003] S. Russell, P. Norvig. *Artificial Intelligence – A Modern Approach*. 2nd edition, Upper Saddle River, New Jersey, USA, Pearson Educations Inc., 2003, 1080 p.

[Willem_2005] F. G. Willem. *Robotics, philosophy and the problems of autonomy*, Pragmatics and Cognition, John Benjamins Publishing, 2005, pp. 515–532.

Dr. Sc. Ing. ***Agris Nikitenko***

Model of autonomous robot

Keywords: *Artificial intelligence, Robotics, Autonomous systems*

The paper is devoted to the project Nr AM 2007/52 funded by Ministry of Defense of Republic of Latvia. The paper describes the main results and problems that were studied within the project.

The main goal of the project was to develop a model of software system that controls autonomous robotic system. The most characterizing feature of autonomous systems is their mobility and ability to plan their actions in unknown and dynamic environments. Therefore most of the project efforts were related to navigation and motion planning problems. The navigation itself consists of several sub-problems including self-localization, mapping and high level planning.

To address all of these problems the well known three-layer architecture ATLANTIS was adopted and slightly modified without loosing the advantages of heterogeneous control in order to fit the technical solution of the software developed within the project. The self-localization is addressed by applying laser scanner data processing algorithm proposed by Martinez. The laser scanner is the leading sensor that is used for robot navigation. Motion planning is addressed by applying RRT planner that is well suited for discreet motion planning. For high level task planning specialized algorithms were built that allow to investigate closed environment and patrol within the closed area by providing several control point on a metric map. The main tool for interacting with user is geographic information system which data is provided by the robotic system thereby addressing the mapping problem.

The robotic system is implemented as 3D visual model.

The further research efforts are related with development of operational prototype, development of appropriate mathematical model and control of continuous motion as well as further advancement of integrated control software for high level task planning.