KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

T125B114

ROBOTŲ PROGRAMAMVIMO TECHNOLOGIJOS

Projektinio darbo ataskaita

**Atliko:**

**Tomas Kašelynas IFF-7/5**

**Giedrius Rastauskas IFF-7/12**

**Lukas Žaromskis IFF-7/5**

**Priėmė:**

**Doc. Brūzgienė Rasa**

**Doc. Adomkus Tomas**

KAUNAS 2020

Turinys

[Užduotis 3](#_Toc58313685)

[Užduoties analizė 3](#_Toc58313686)

[Roboto aprašymas 3](#_Toc58313687)

[Roboto valdymo architektūra 4](#_Toc58313688)

[Roboto valdymo algoritmas 4](#_Toc58313689)

[Roboto modeliavimo rezultatai 5](#_Toc58313690)

[Roboto valdymo programa 6](#_Toc58313691)

[Roboto valdymo eksperimentinis tyrimas 7](#_Toc58313692)

[Išvados 7](#_Toc58313693)

[Naudota literatūra 8](#_Toc58313694)

# Užduotis

Robotas važiuoja link labirinto pradžios. Aptikęs kliūtis, jas apvažiuoja BUG0 algoritmu. Labirintą įveikia pagal dešinės rankos taisyklę. Toliau važiuoja link pabaigos taško ir kliūtis apvažiuoja ??? algoritmu.

# Užduoties analizė

Roboto pradinis taškas yra toliau nuo labirinto ir jis turi pasiekti jo pradžią. Važiuoti tiesiai robotui nepavyks, nes tarp jo ir labirinto pradžios bus kliūčių. Kai robotas pasieks labirintą, jis jį turės įveikti ir pasiekti jo pabaigą.

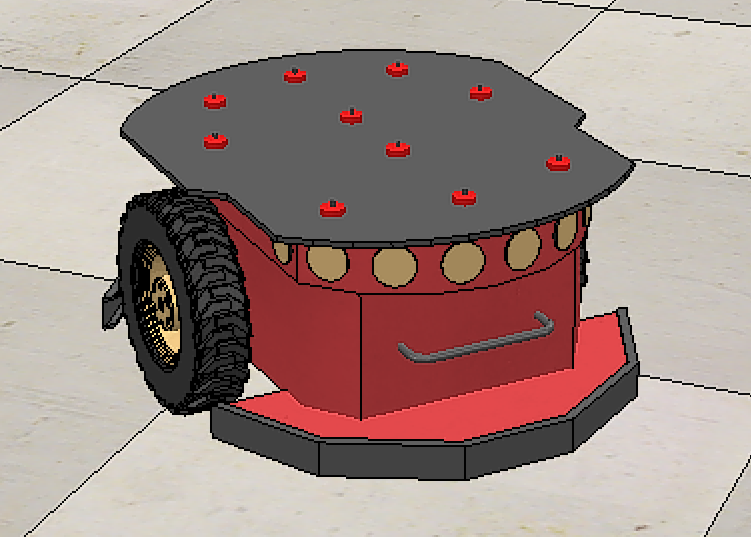
Reikalinga, kad robotas mokėtų apvažiuoti kliūtis ir įveikti labirintą pagal dešinės rankos taisyklę.

# Roboto aprašymas

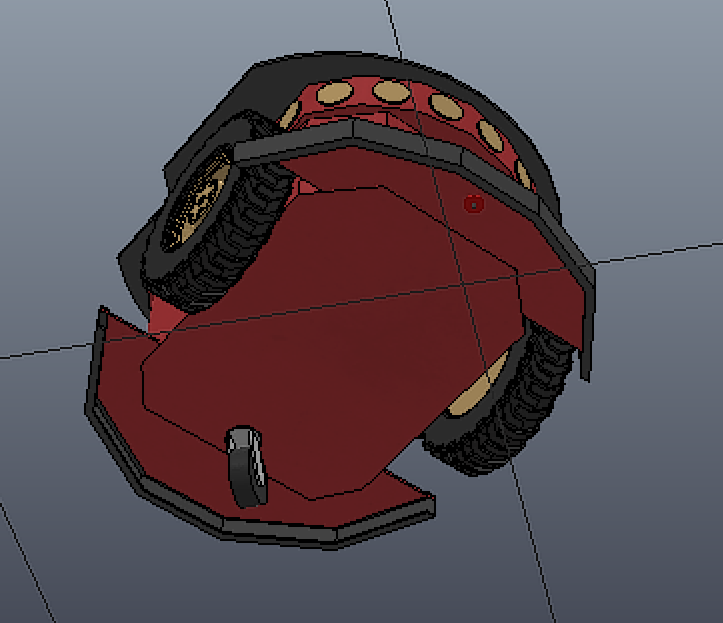
Robotas turi tris ratus: du pagrindiniai ratai su motorais ir vienas ratukas stabilizacijai. Taip pat robotas turi 16 ultragarsinių atstumo sensorių bei 11 jungčių.



1 pav. Roboto priekis



2 pav. Roboto galas



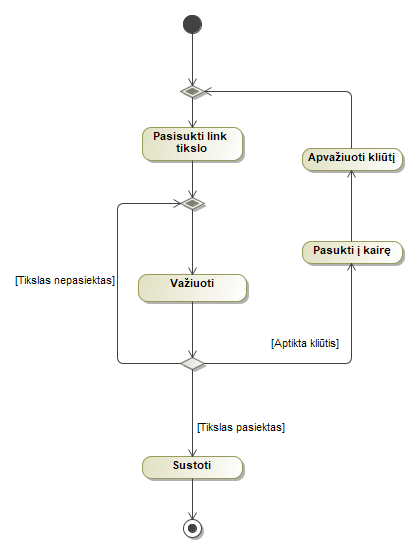
3 pav. Roboto apačia

# Roboto valdymo architektūra

Robotas valdomas siunčiant jam komandas per Python „sim“ API [1]. Galima nustatyti kairiojo arba dešiniojo rato sukimosi greičius, gauti informaciją iš sensorių.

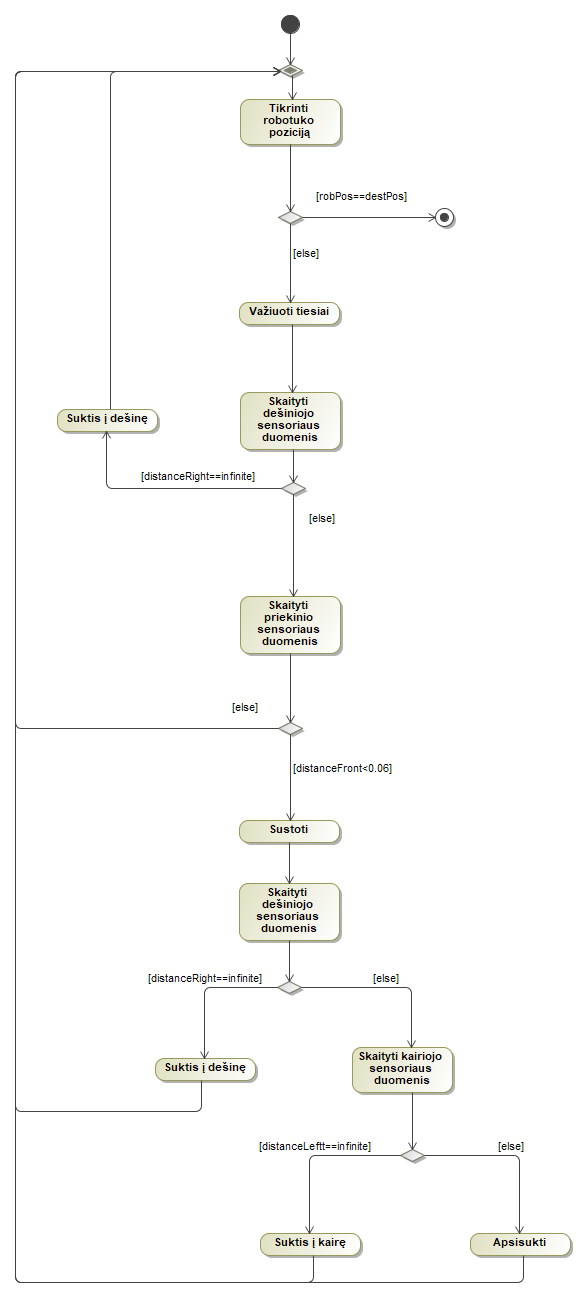
# Roboto valdymo algoritmas

Pirmąsias kliūtis robotas įveikia naudojant BUG0 algoritmą. Jis važiuoja tiesiai link tikslo pozicijos, o kai yra aptinkama kliūtis yra sukama į kairę ir kliūtis apvažiuojama. Toliau vėl yra važiuojama tiesiai link kliūties ir jei yra aptinkama kliūtis, yra kartojamas apvažiavimas.



4 pav. BUG0 veiklos diagrama

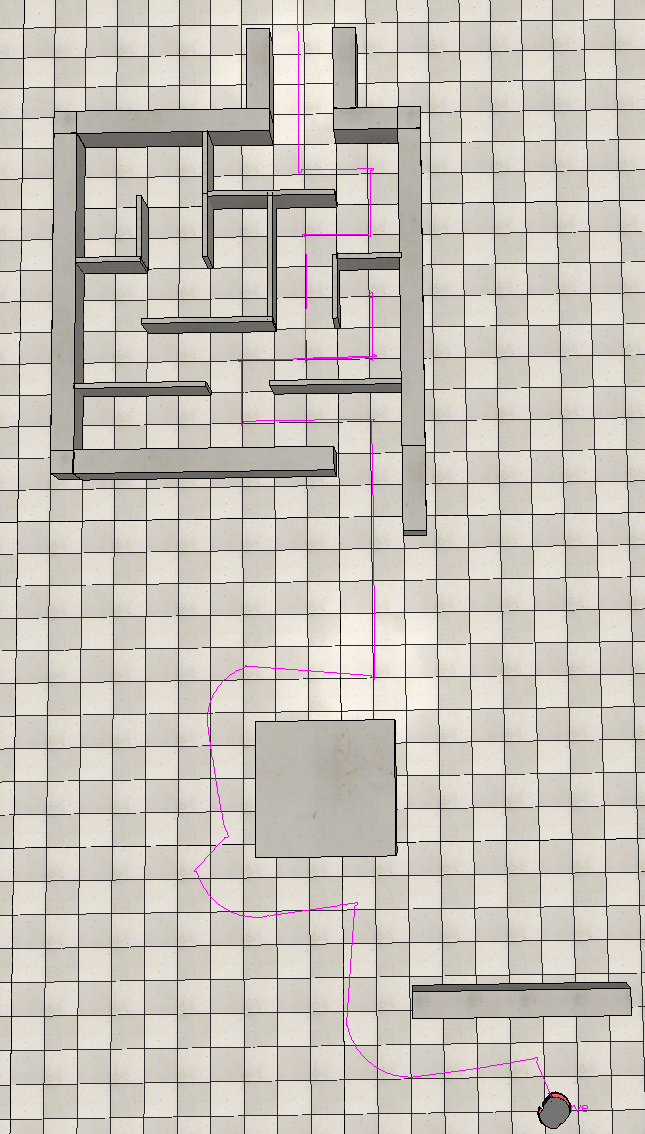
Labirintą robotas įveikia naudojant dešinės rankos taisyklės algoritmą. Robotas važiuoja tiesiai tol, kol išvažiuoja iš labirinto. Aptikęs priešais sieną, pasižiūri, ar gali važiuoti į dešinę pusę. Jei gali, pasisuka į dešinę ir važiuoja tiesiai. Jei negali, tikrina kairę pusę. Jei ten nėra sienos, sukasi į kairę ir važiuoja tiesiai. Jei negali važiuoti nei į dešinę, nei į kairę, apsisuka ir važiuoja tiesiai. Jeigu važiuojant tiesiai aptinka, kad gali važiuoti į dešinę, sukasi į dešinę ir važiuoja tiesiai.



5 pav. MazeSolver veiklos diagrama

# Roboto modeliavimo rezultatai

Simuliacijos rezultatas pateiktas 6 pav. Robotukas yra pradinėje pozicijoje. Rožinė linija rodo jo judėjimo trajektoriją. Jo trajektorija rodo, kad jis juda link tikslo tol, kol pasiekia kliūtį. Tada ją apvažiuoja ir bando toliau judėti link tikslo. Kartoja tol, kol pasiekia tikslą. Tada važiuoja link labirinto, kurį įveikia pagal dešinės rankos taisyklę – seka dešinę sieną.



6 pav. Simuliacijos rezultatai

# Roboto valdymo programa

# Roboto valdymo eksperimentinis tyrimas

Robotas gali įveikti bet kokį įveikiamą labirintą pagal dešinės rankos taisyklę, tačiau labirinto sienos turi būti lygiagrečios x ir y ašims, kad robotukas tiksliai pasisuktų 90 laipsnių kampu. Užduotis yra įveikiama šiek tiek greičiau nei per 8 minutes. Daugiausiai laiko užtrunka 90 laipsnių pasisukimai dėl duoto mažo sukimosi greičio, kad pasisukimai butų kuo tikslesni.

# Išvados

1. Tomo Kašelyno (labirinto įveikimas pagal dešinės rankos taisyklę, labirinto pastatymas simuliacijoje, testavimas, ataskaitos pildymas) išvados:
   1. Ne visus labirintus galima įveikti pagal dešinės rankos taisyklę.
   2. Svarbu užtikrinti, kad robotukas judėtų kuo lygiagrečiau labirinto sienoms, nes kitaip gali aptikti sieną ir pradėti suktis netinkamu laiku.
   3. Norint įveikti labirintą pagal dešinės rankos taisyklę, robotukas turi teikti prioritetą pasisukimams į dešinę.
   4. Aptikęs galimą pasisukimą į dešinę, robotukas turi iš karto suktis į dešinę.
2. Luko Žaromskio (bug0 algoritmo realizacija, 90 laipsnių pasisukimo realizacija, kliūčių pridėjimas iki labirinto, testavimas, ataskaitos pildymas) išvados:
3. Giedriaus Rastausko išvados:

# Naudota literatūra

|  |  |
| --- | --- |
| [1] | „Coppelia Robotics Remote API functions (Python),“ [Tinkle]. Available: https://www.coppeliarobotics.com/helpFiles/en/remoteApiFunctionsPython.htm. [Kreiptasi 8 December 2020]. |