Universidad del País Vasco Euskal Herriko Unibertsitatea
GIPUZKOAKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE GIPUZKOA

Industria Elektronikaren eta Automatikaren Ingeniaritzako Gradua

ERANSKINA 4:

ROS 2ren OINARRIZKO GIDA IV

SUMMIT ROBOTAREN ERABILTZAILEAREN ESKULIBURUA ROS 2n

Ikaslearen izen eta abizenak: Iñaki Azkarragaurizar Jauregi

Irakaslearen izen eta abizenak: Mikel Larrea Sukia

Data: Donostia, 2023ko uztailaren 21a

Aurkibidea:

Argazkien Zerrenda

Komandoen Zerrenda

1. SARRERA

Dokumentu hau, gradu amaierako lanean garatu diren funtzionaltasunei buruzko erreferentzia bat sortzeko sortu da. Gida honetan, Robotnik enpresaren Summit robota, ROS 2 softwarearen [1] bitartez simulazio-ingurune batean martxan jarri nahi duen edozein erabiltzaileri zuzenduta dago, robotaren nabigazio autonomoa gaitzen duten konfigurazioekin. Abian jartzeaz gain, eskatzen diren oinarrizko ezagutza teknikoak azaltzen dira, hala nola arkitekturaren, nodoen konexioen eta zeinbat konfigurazioen azalpenak. Guzti hau, publiko ororentzat balio dezan egin da, ezagutza teknikorik ez dituenak kontuan hartuz.

Erabiltzailearen eskuliburu hau sortu aurretik, beste hiru gida sortu dira, non bertan, erabiliko den softwarearen oinarrizko ezagutzak azaldu eta bi robot mota desberdin zerotik sortzeko argibideak adierazten dira. Dokumentu hau, Summit proiektua ostatzen duen GitHub-eko gordailuan oinarritzen da [2], bertan erabiliko diren fitxategi eta programazio-kode guztiak aurki daitezke.

1.1 Robotnikeko Summit Robota.

Gida honek Summit-aren aplikazioak simulazio-ingurune batean martxan jartzeko balio du, eta, horrekin batera, dokumentuen azalpen zehatzagoa ematen du, gradu amaierako lanaren memorian osagarri bezala jokatuz.

Robotnik enpresak Summit plataforma mugikor bat eskaintzen du, Ackermann mugimendu-sistema erabiltzen duena, hau da, biraketaren ezaugarriak eta geometria auto batenak dira. Robot horren erabilera nagusia esparru akademikoa eta ikerketa-eremua dira, kasu honetan bezala, GIEk horietako bat baitu ikerketa-lanak gauzatzeko.

1. irudia: Robotnikeko Summit robota [3]

Robot hau bateragarria da ROSeko software aurreratuarekin, ordenagailu bat daramalako (Intel D945GSEJT). Horretaz gain, Wi-fi modulu bat ere badu, beste PC batzuekin komunikatzeko aukera ematen duena.

Atal honi buruzko informazio gehiago GrALaren memorian edo, behar izanez gero, Robotnikeko *datasheet*-etatik [3] lor daiteke.

1.2 Aurrebaldintzak.

Gomendagarria:

Erabiltzailearen eskuliburu hau, ez dago beste gida baten mende, dokumentu honetan azaltzen baita garrantzitsua den guztia. Baina aurreko gidetan The Construct plataformari [4] buruzko beharrezko ezagutza guztia azaldu da, ROS 2ren oinarrizko nozioak, dokumentuak nola sortu, konfigurazioen azalpena... Hori guztia ez da azalduko gida honetan, eta, beraz, beharrezkoa izanez gero, dokumentu hauek kontsultatu:

- ROS 2ren Oinarrizko Gida I.
- ROS 2ren Oinarrizko Gida II.
- ROS 2ren Oinarrizko Gida III.

Bestalde, ROS 2 Humble Hawksbill bertsioa erabilitzeko, The Construct plataforma erabiliko da, honek ROSeko simulazio-ingurune bat eskaintzen baituelako.

Beharrezkoa:

Aurreko gidetan bezala, dokumentu honetan ere The Construct plataforma erabiltzen da, robotentzako ROS 2 Humble Hawksbill-aren bertsioan, simulazio-ingurune bat eskaintzen baituelako. Horri esker, gida honetan erabiliko diren pakete asko deskargatuta egongo dira. Bertan, soilik, eskuragarri dauden paketeen zerrenda eguneratu eta sisteman instalatutako paketeak eguneratu beharko dira.

Horretarako terminal bat ireki:

```
sudo apt update
```

1. Komandoa: eskuragarri dauden paketeen zerrenda eguneratzeko komandoa

Komando hau exekutatzean, sistemak instalatutako paketeetarako eguneratze erabilgarriak bilatuko ditu, eta tokiko biltegien informazioa eguneratuko du.

Honen ostean, terminal berdinean hurrengoa exektutatu behar da:

```
sudo apt-get upgrade
```

2. Komandoa: eskuragarri dauden paketeen zerrenda eguneratzeko komandoa

Sisteman instalatutako paketeak eguneratzeko erabiltzen da komando hau. Horri esker, plataforma honetan nabigazioko liburutegiak, beste zenbaiten artean erabiltzea ahalbidetuko du.

The Construct plataformari buruzko informazio gehiago, GrALaren memorian aurki daiteke edo bere webgune ofizialean [4].

2. ERABILTZAILEAREN ESKULIBURUA

Atal honetan, ingurune simulatu batean robota martxan jartzeko egin beharreko urratsei buruzko gida zehatza emango da. Hiru atal nagusitan banatuko da: Summit-a simulazio-ingurunean abiarazi, teklatu bidezko kontrola eta nabigazio autonomoa.

Puntu honekin hasteko, beharrezkoa da "aurrekobaldintzak" atala osatua izana, beharrezko paketeak instalatuta eta eguneratuta edukitzeko.

Guztiz funtzionala den Summit robotaren gordailua, GitHub plataforman aurkitzen da, eta ordenagailuan kopiatzeko, komando hurrengoa da:

```
git clone https://github.com/lukin1225/inaki_ws.git
```

3. Komandoa: GitHub plataformatik, Summit-aren gordailua kopiatzeko komandoa.

Kopiatutako pakete guztiak konpilatzeko, lan-espazioaren barruan egin behar da, horretarako:

```
cd ~/inaki_ws/
colcon build
```

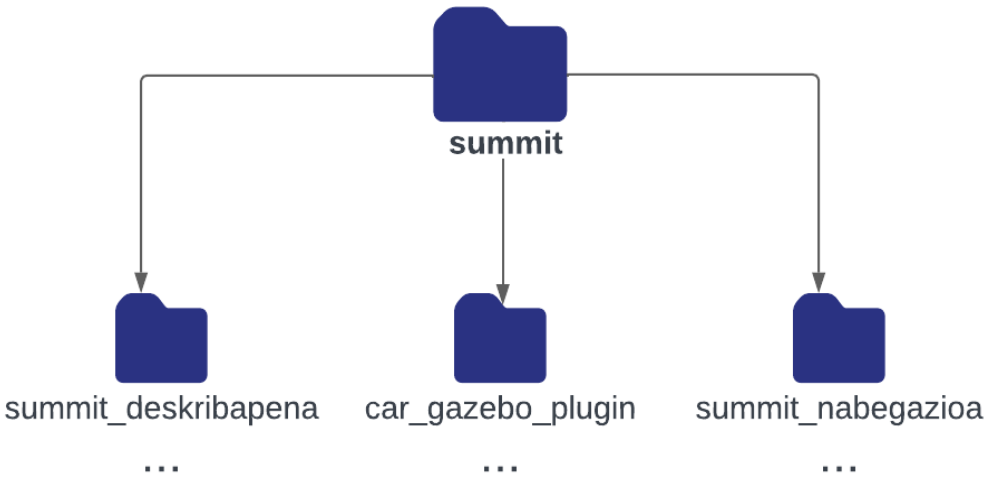
4. Komandoa: paketea konpilatzeko komandoa.

Eta lan-ingurunean instalatutako paketeak ezagutu eta eskuratu ahal izateko, komando hau exekutatu:

```
source install/setup.bash
```

5. Komandoa: lan-espazioa kargatzeko komandoa.

Urrats horiek egin ondoren, biltegian dauden pakete guztiak kopiatuta eta konpilatuta egon beharko lirateke. Gordailu horren egitura 2. irudian ikus daiteke.



2. irudia: Summit proiektuaren paketeen egitura.

2. irudian ikus daitekeen bezala, Summit karpetak hiru pakete desberdinez osatuta dagoela nabarmentzen da. Horietako bakoitzak atal bat betetzen du:

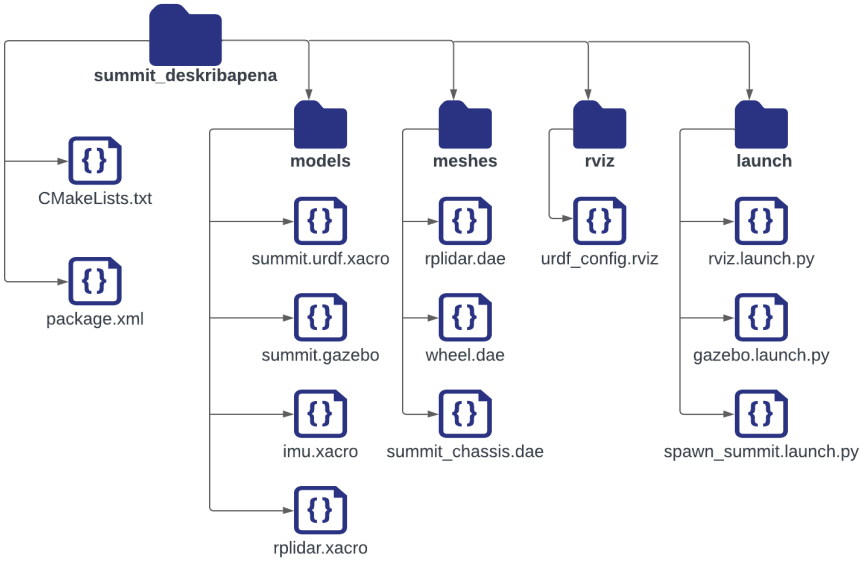
- summit_deskribapena: Robota, RViz eta/edo Gazebo tresnaren bitartez birtualki simulatzea.

- car_gazebo_plugin: Robotaren mugimendu-sistema kontrolatzea.

- summit_nabegazioa: Ibilgailua nabigazio autonomoarekin inplementatzea.

2.1 Summit-a Simulazio-Ingurunean Abiarazi

Robota simulazio-eremuan bistaratzeko, summit_deskribapena izeneko paketea arduratzen da horretaz. Bertan, beharrezko karpeta, fitxategi eta konfigurazio desberdinak aurki daizteke (3. irudian ikusgai). Fitxategi hauen edukien azalpen teorikoa, memoriarekin batera atxikituko den 2. eta 3. gidetan aurki daiteke.



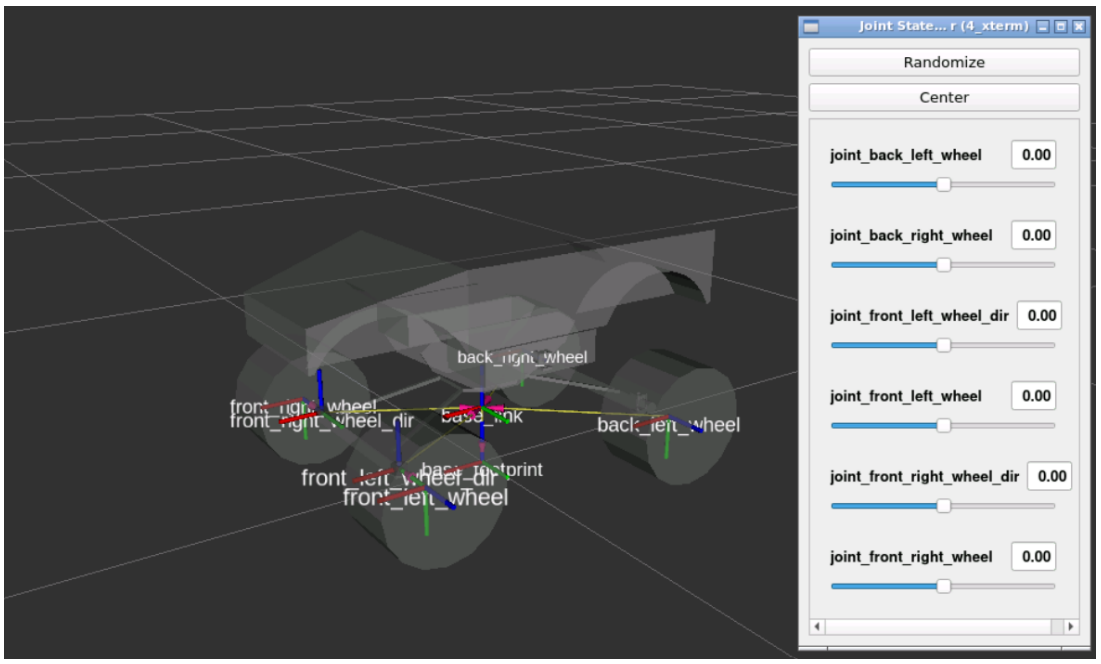
3. irudia: summit_deskribapena paketeak duen egituraren ikusizko irudikapena

Simulzio bat abian jartzeko lehenengo pausua beti, kodea konpilatuta egotea da, pausu hau aurreko atalean egin da.

Summit-a RVizen:

```
ros2 launch summit_deskribapena rviz.launch.py
```

6. Komandoa: simulazioa RViz tresnan exekutatzeko komandoa.



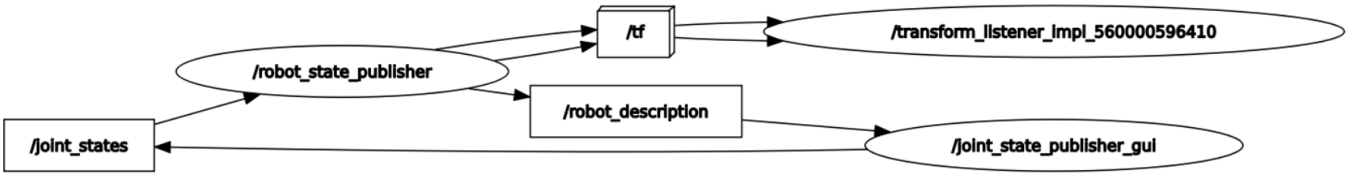
4. irudia: RViz simulazio-ingurunean ikusitako modelo simulatua

Komando hori exekutatzean, Summit-a abiaraziko du RViz ingurunean, robotaren zinematika ikustea ahalbidetzen duen nodo batekin batera, “start_joint_state_publisher_gui_node”. Nodo honekin jolastuz, sortutako loturei balioak eman ahal zaie, hala nola aurreko gurpilak biraraziz edo aurreko gurpilen norabideen baloreak aldatuz, gurpilak angelu batean biraraziko dute. Horri esker, robotak duen Ackermann sistema-mugimendua hobeto uler ahal izango da.

Exekutatu den *launch* fitxategiarekin, abiarazi diren nodoen arteko konexioa era grafiko batean ikusteko, hurrengo komandoa exekutatu:

```
rqt_graph
```

7. Komandoa: nodoen konexioak era grafiko batean ikusteko komandoa.



5. irudia: rqt_graph-en bitartez nodoen konexioak era grafiko batean

Goiko irudi honetan (5. irudia) ikusi daiteke nola "/joint_state_publisher_gui" nodoak "/joint_state" gaiean datuak (gurpilaren mugimenduak) publikatzen dituen. Datu horiek, "/robot_state_publisher" nodoak jasotzen ditu eta koordenatu-marko horien informazioa "tfn" argitaratzen ditu. Eta horrela, RVizen roboteko gurpilen posizio berria eguneratzen da. Oraintxe azladu dena, "start_joint_state_publisher_gui_node" nodoarekin jolastean ibiltzearen, gertatzen denaren azalpena da.

Summit-a Gazebon:

Summit-a Gazebo simulazio-eremuan abiarazteko beste *shell* bat ireki beharko da, eta terminal horrek instalatutako paketeak ezagutu eta eskuratu ahal izateko, komando hau berriz exekutatu:

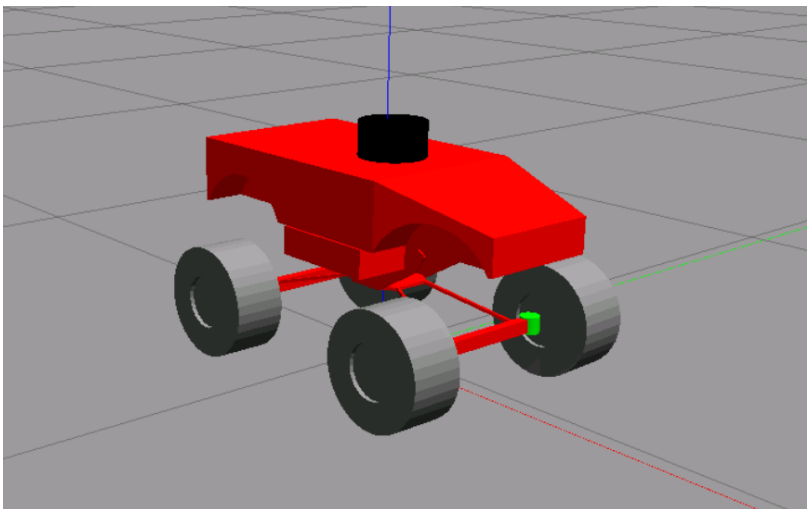
```
source install/setup.bash
```

Oharra: komando hau terminal berri bat irekitzen denean exekutatu behar da beti. Gainera workspace-aren barruan egin behar da beti.

Ondoren terminal berdinean, jaurtiketa-fitxategi hau abiarazi:

```
ros2 launch summit_deskribapena gazebo.launch.py
```

8. Komandoa: simulazioa Gazebo tresnan exekutatzeko komandoa.

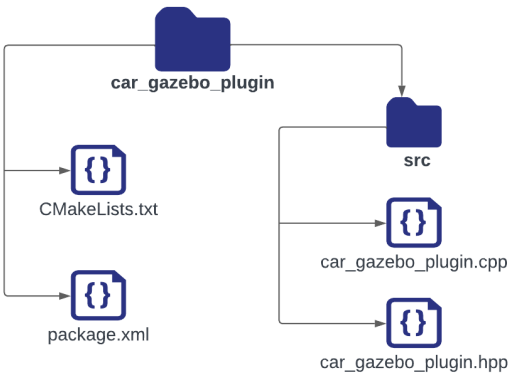


6. irudia: Gazebo simulazio-ingurunean ikusitako modelo simulatua

Komando hori exekutatzean, Summit-a Gazeboko mundu huts batean abiaraziko da. Horretaz gain, RVizen ere bistaratuko du, baina kasu honetan robataren zinematika ikustea ahalbidetzen duen nodoa gabe

2.2 Teklatu Bidezko Kontrola

Robota simulazio-eremuan bistaratu ondoren, car_gazebo_plugin izeneko paketea erabiliko da Summit-aren kontrola ahalbidetzeko. Aipatutakoa gauzatzeko, karpeta horrek beharrezko pakete, fitxategi eta konfigurazio desberdinak ditu bere barruan (7. irudian ikusgai). Fitxategi hauen edukien azalpen teorikoa, memoriarekin batera atxikituko den 3. gidan aurki daiteke.



7. irudia: car_gazebo_plugin paketeak duen egituraren ikusizko irudikapena

Simulazioa martxan dagoenean (x irudian bezala), "teleop_twist_keyboard" nodoa exekutatu daiteke, teklatuaren bidez robota eskuz kontrolatu ahal izateko. Horretarako, terminal berri bat ireki eta artxibo hau exekutatuko da:

```
ros2 launch summit_deskribapena gazebo.launch.py
```

9. Komandoa: robot baten mugimendua teklatuaren bidez kontrolatzeko komandoa

Komando hori exekutatzean, "teleop_twist_keyboard" nodoa abiarazten da, eta, horri esker, erabiltzaileak abiadura linealeko eta angeluarreko komandoak bidal ditzake robotera, teklatuko teklak erabiliz. Esandakoa hobeto ulertzeko terminal berri batean hurrengoa exekutatu:

```
rqt_graph
```

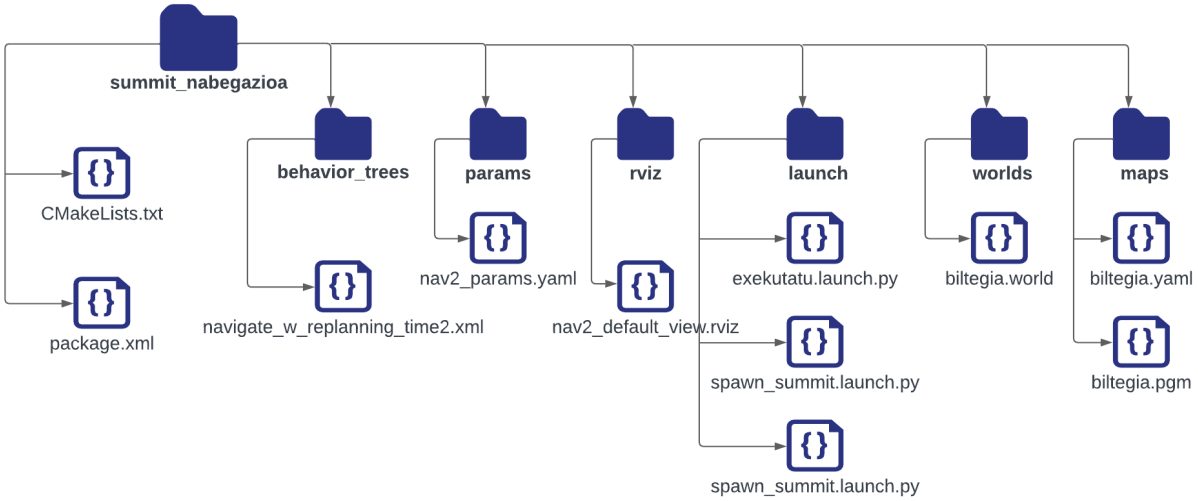
8. Irudia: nodoen arteko konexioen irudikapena.

Goiko iruditik hurrengoa interpretatu daiteke: "teleop_twist_keyboard" nodoak abiadura linealeko eta angeluarreko komandoak bidal ditzake, teklatuko teklak erabiliz. Hau da, nodoak teklatuko sarrerak interpretatzen ditu eta dagozkion datuak /cmd_vel gai-eremuan argitaratzen ditu. Ondoren, atal honetan gehitutako pluginari esker, car_gazebo_plugin nodoa /cmd_vel gaiera harpidetzen da, eta berariazko komandoak bihurtzen ditu Ackermann motako sistema duten ibilgailuentzat.

Bukatzeko, robotaren artikulazioen egoerari buruzko informazioa /joint_states gaian argitaratzen da eta honi /robot_state_publisher nodoa izenpetzen zaio. Nodo honek robotaren egoera argitaratzen du "URDF" izeneko formatu batean, eta, horrela, Summit robotaren simulazioa eta urrutiko kontrola ahalbidetzea lortzen da.

2.3 Nabigazio autonomoa

Robota simulazio-eremuan bistaratu eta teklatu bidezko kontrola gehitu ondoren, summit_nabegazio izeneko paketea erabiliko da, simulazio-eremuan Summit-a era autonomo batean nabigatu dezan. Aipatutakoa gauzatzeko, karpeta horrek beharrezko pakete, fitxategi eta konfigurazio desberdinak ditu bere barruan (9. irudian ikusgai). Fitxategi hauen edukien azalpen teorikoa, memoriarekin batera atzikituko den 3. gidan aurki daiteke.



9. Irudia: summit_nabigazioa paketeak duen egituraren ikusizko irudikapena

Nabigazioarekin hasteko, aurretik exekutatutako nodoekin amaitu behar dira, berriro abiaraziko direnez, elkarren artean ez gainjartzeko. Pausu hau, 2.1 eta 2.2 ataleko komandoak abiarazi badira soilok egin behar da. Horretarako terminal berri bat ireki eta hurrengo komandoa idatzi:

```
killall5
```

10. Komandoa: gauzatzen ari diren prozesu guztiak amaitu eta sistema berrabiarazten duen komandoa.

Ondoren, hurrengo komandoa exekutatu *workspace*-ean.

```
source install/setup.bash
```

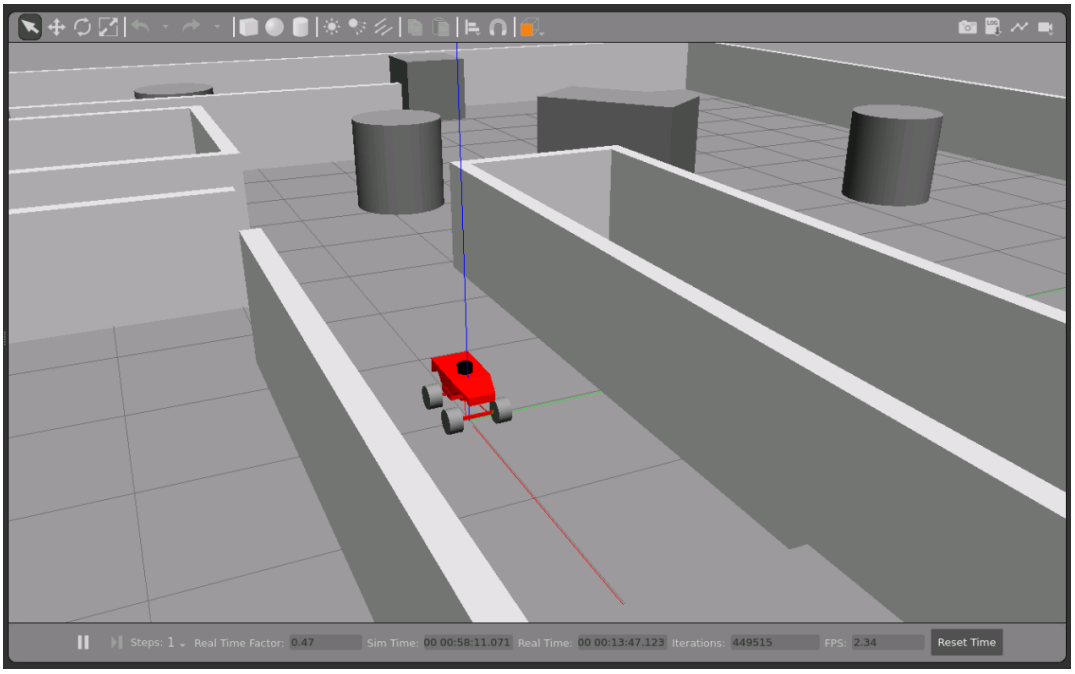
Puntu honetan, Summit-a Gazeboko simulazio-eremuan abiarazi daiteke, honen nabigazio autonomoa ahalbidetzen dituen konfigurazio guztiekin, azkenengo hau RVizen bitartez gobernatuko da. Horretarako hurrengo komandoa exekutatu:

```
ros2 launch summit_nabegazioa exekutatu.launch.py
```

11. Komandoa: Summit-a nabigazio autonomoarekin exekutatzeko komandoa.

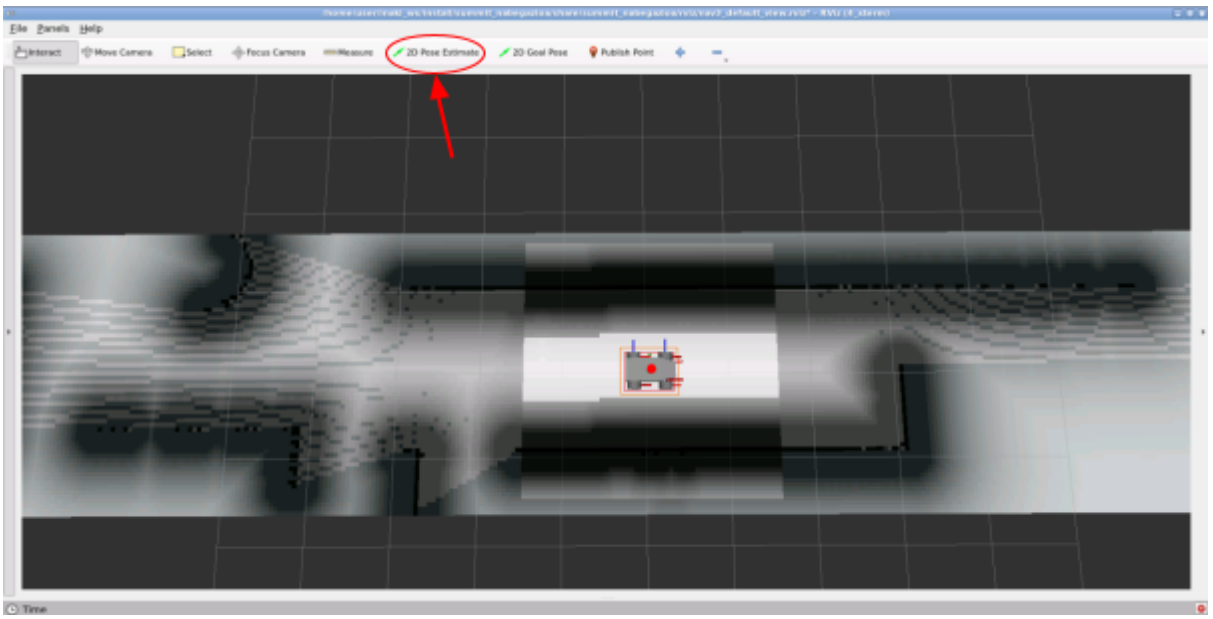
Komando hau exekutatzean bi leiho ireki beharko lirateke:

1. Summit robota, Gazeboko ingurune-simulatu batean (10. irudian ikusgai).
2. Robot bera baina RVIz tresnan abiarazita, beharrezkonabigazio konfigurazio guztiekin batera (11. irudian ikusgai).



10. Irudia: Summit-a Gazebon abiarazita, biltegia izeneko ingurune-simulatuan

Goiko irudian ikus daitekeen bezala, Summit-a Gazeboko ingurune batean ageri da, biltegia.world fitxategiari esker.

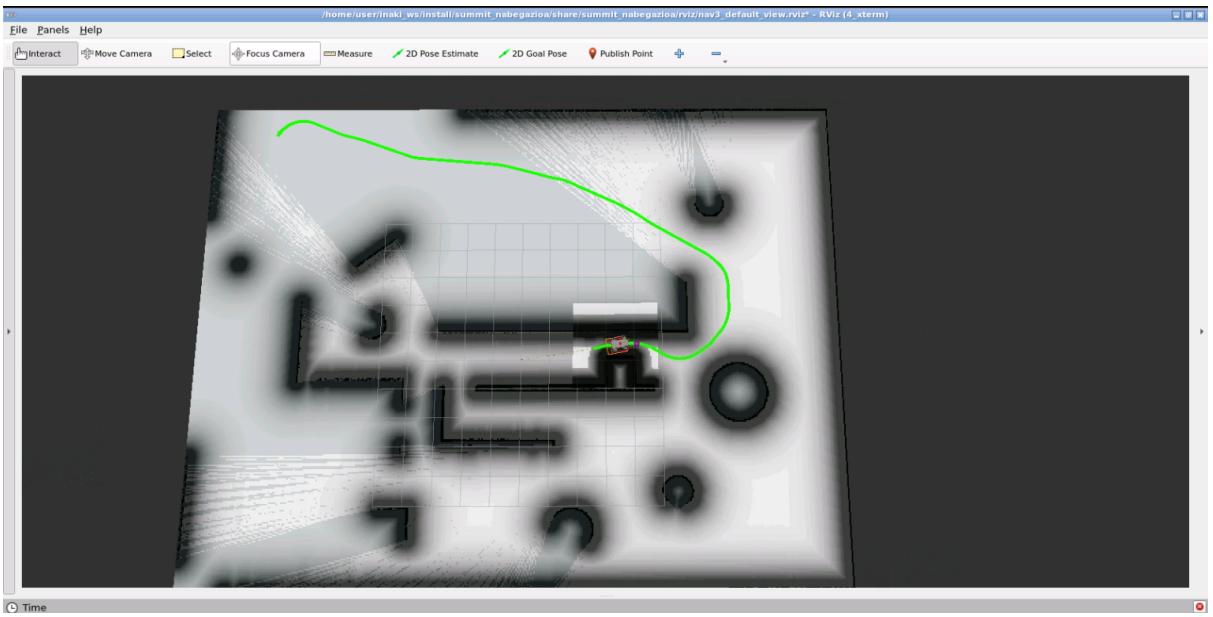


11. Irudia: Summit-a RVizen abiarazita, nabigaziorako

Irekitzen den bigarren leihoari dagokionez, robota RViz ingurune-simulatuan bistaratzen da, nabigazio autonomoa ahalbidetzen duten konfigurazio guztiekin. Kasu honetan, nabigazioaren zereginarekin osatzeko, SLAM metodoa erabili da (Simultaneous Localization And Mapping). Teknika hau, robota ingurune ezezagun bat ikertu eta mapatzeko erabiltzen da, aldi berean sortutako mapa horren barruan aurkitzen den bitartean.

Oharra: baliteke RViz irekitzean nabigaziorako beharrezko datuak eta konfigurazioak kargatu ez izana. Ziurtatu x. argazkian ikusten denaren berdina izatea, horrela ez bada itxaron minutu batzuk.

Nabigatzeko, 11. irudiaren goikaldean ageri den “2D Goal Pose” izeneko tresna erabiltzen da. Tresna horri esker, mapan helmuga bat ezar daiteke, robota puntu horretara bideratuz, 12. irudian ikusgai



12. Irudia: Summit robota bere helmuga berrira joaten

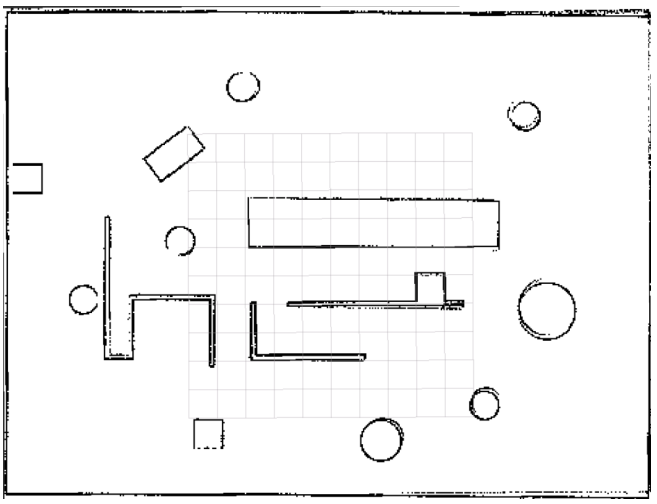
Argazkian ikus daitekeenez, robotari helmuga zein den adierazten zaionean, hasierako ibilbide bat marratzen du (lerro berdea). Aipatu den bezala, SLAM teknika erabiliz nabigatzen ari da, eta robotak hasieran ez daki bidean aurkituko dituen oztopoak. Baina ibilbidea 1 hz-ko aldian eguneratuz doa, eta laser-sentsore bat txertatuta daramanez, oztopo bat detektatzen duenean ibilbidea birbideratu egingo du, talka saihestuz. Azaldu berri dena nola lortu den, konfigurazio aipagarrien atalean (3.2) aurki daiteke.

Mapa estatikoa sortu

Lehenago aipatu den bezala, nabigazio autonomoa SLAM teknikaren bitartez egitean, robota hainbat lekutatik aurrera egin ahala ingurunearen mapa bat sortzen joango da. Beraz, robotak simulazio-inguruneko txoko guztietan nabigatu ondoren, sortutako mapa gordetzeko, hurrengo komandoa exekutatu:

```
ros2 run nav2_map_server map_saver_cli -f biltegia
```

12. Komandoa: mapa estatiko bat sortzeko komandoa.



13. irudia: RViz programan LIDAR sentsoreak sortutako mapa

Horrela, sortutako mapa estatikoa gorde da eta biltegia izena jarri zaio. Komando hau exekutatzean bi fitxategi sortuko dira: biltegia.pgm eta biltegia.yaml.

3. GEHIGARRIAK

3.1 Interpretazio Tresnak

ROS software konpleto bat da, eta hainbat tresna eskaintzen dizkio erabiltzaileari ingurunean gertatzen ari dena interpretatzeko eta dauden kode akatsak zuzentzeko. Horien artean lehen erabili den tresna bat aurkitzen da, rqt_graph. Horri esker, nodo aktiboen konexioak oso modu grafikoan ikus daitezke. Horrek ROS 2 sistemaren arkitektura ulertzea errazten du, eta komunikazio-arazo posibleak identifikatzen laguntzen du.

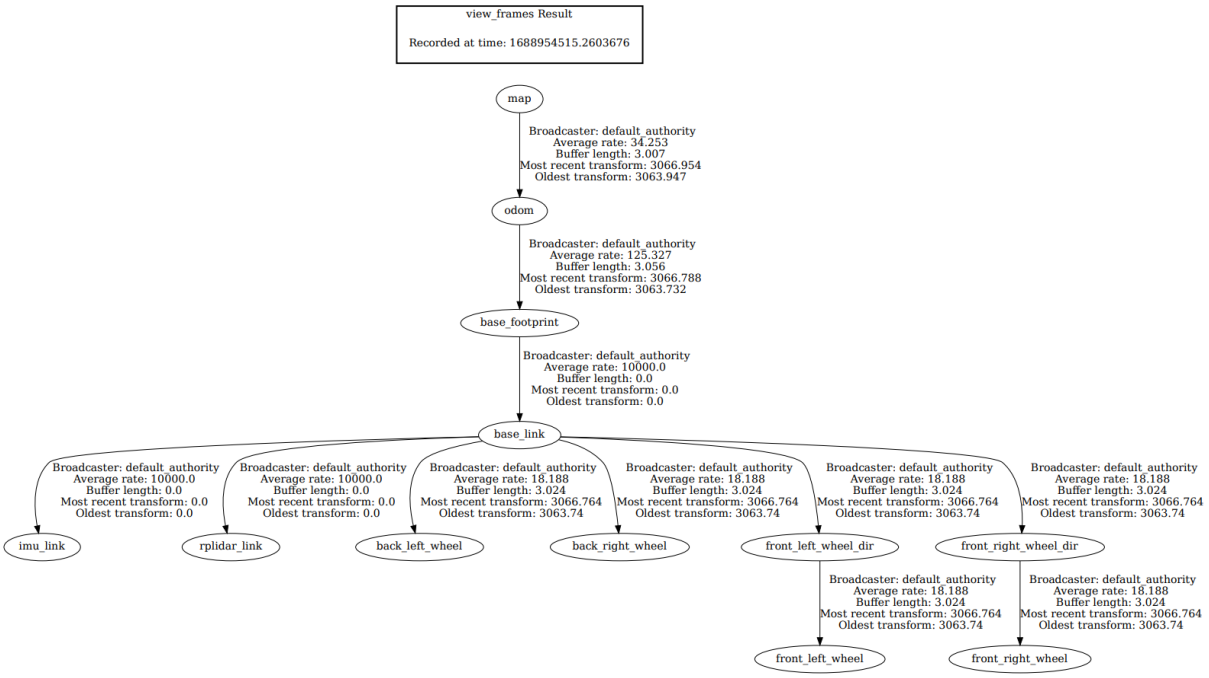
rqt_graph

Komando hori exekutatu ondoren, nodoen eta topikoen konexioak modu argi eta bisualean erakusten duen grafiko bat bistaratzen da. Horren irudia proiektuarekin batera gehituko da (5. eranskina), oso handia baita eta dokumentu honetan ikusi ezin diren konexio asko baititu.

ROS 2k eskaintzen duen beste tresna interesgarri bat, hurrengoa da:

```
ros2 run tf2_tools view_frames
```

13. Komandoa: exekutagarri bat abiarazteko komandoa.



14. irudia: Summit-aren transformatuen zuhaitza.

Komando hori exekutatzean, erreferentzia-markoen (*frames*) arteko transformazioen hierarkia islatzen du. Eraldaketa horiei esker, sistemaren osagaiek (lidar-sentsorea, gurpilak, etab.) maparen erreferentzia esparru orokorrean lan egiten dute, eta elkarren artean komunikatu. Hori funtsezkoa da nabigazio zehatza egiteko eta inguruneko ibilbideak planifikatzeko.

3.2 Konfigurazio Aipagarriak

Nabigazio autonomoaren atalean, 12. irudiari buruz azaldu dena lortzeko, hiru konfigurazio garrantzitsu bereziki har daitezke kontuan, nahiz eta nabigazio osoa parametro askoz gehiagoren mende egon:

1. **Behavior Ttrees:**

Portaera Zuhaitzak, robotak kontrolatzeko eta koordinatzeko erabiltzen dira, modu sekuentzialean exekutatzen baitira, nodo bakoitza zehaztutako baldintzen eta irizpideen arabera ebaluatuz.

Beraz, robotak 1 hz-ko aldian ibilbide globala birplanifikatzearen ekintza, Behavior Tree bati esker lortzen da. Horretarako, "navigate_w_replanning_time2.xml" izeneko fitxategi bat erabiltzen da (9. irudian ikusgai), aipatu berri dena gauzatzeko.

navigate_w_replanning_time2.xml:

```
<root main_tree_to_execute="MainTree">
  <BehaviorTree ID="MainTree">
    <PipelineSequence name="NavigateWithReplanning">
      <RateController hz="0.1">
        <ComputePathToPose goal="{goal}" path="{path}" planner_id="GridBased"/>
      </RateController>
      <FollowPath path="{path}"  controller_id="FollowPath"/>
    </PipelineSequence>
  </BehaviorTree>
</root>
```

1. scripta: navigate_w_replanning_time2.xml

Honi buruzko informazio gehiago, Nav2 webguneko dokumentazio ofizialean aurki daiteke [5].

2. **Ibilbidearen Planifikatzailea:**

12. irudian ikusten den kolore berdeko ibilbidea marratu ahal izateko, proiektuko nav2_params.yaml fitxategian (9. irudian ikusgai), plugin espezifiko bat erabili behar izan da. Hau da, ibilbidearen planifikatzaile gisa, "nav2_smac_planner/SmacPlannerHybrid" plugina erabiltzen da eta bere konfigurazioa hurrengoa da:

nav2_params.yaml (kode zati bat):

```
GridBased:
  plugin: "nav2_smac_planner/SmacPlannerHybrid"
  downsample_costmap: false
  downsampling_factor: 1
  allow_unknown: true
  max_iterations: 1000000
  max_planning_time: 5.0
  motion_model_for_search: "REEDS_SHEPP"
  angle_quantization_bins: 72
  analytic_expansion_ratio: 3.5
  analytic_expansion_max_length: 3.0
  minimum_turning_radius: 1.0
  reverse_penalty: 50.0
  change_penalty: 0.0
  non_straight_penalty: 1.2
  cost_penalty: 3.0
  lookup_table_size: 20.0
  cache_obstacle_heuristic: false
  smooth_path: False
```

2. scripta: nav2_params.yaml

Erakutsitako kodeari esker, bide eraginkorrak bermatu eta inguruko oztopoak saihestuko dituzten ibilbideak planifikatuko ditu robotak. Robota atzeraka mugitu ahal izateko, parametro garrantzitsuenetako bat kodeko 8. lerroan aurkitzen da, "REEDS_SHEPP". Parametro horri esker, planifikatzaileak kontuan hartzen ditu erabilitako ibilgailuak dituen mugimendu-murrizketak, eta, horrela, atzerako ibilbideak marratzea lortzen du. Gainontzeko parametroen konfigurazioei dagokionez, Nav2 webguneko dokumentazio ofizialean aurki daiteke [5].

3. Nabigazio-kontrolatzailea:

Aurreko puntuan planifikatutako ibilbidea era zehatz eta leun baten jarraitzeko helburua betetzeko, "FollowPath" nabigazio-kontrolatzailea, RPP pluginarekin erabili da. Bere konfigurazioa hurrengoa da:

nav2_params.yaml (kode zati bat):

```
  FollowPath:
    plugin: "nav2_regulated_pure_pursuit_controller::RegulatedPurePursuitController"
    allow_reversing: true
    approach_velocity_scaling_dist: 0.6
    cost_scaling_dist: 0.6
    cost_scaling_gain: 1.0
    desired_linear_vel: 0.5
    inflation_cost_scaling_factor: 3.0
    lookahead_dist: 0.6
    lookahead_time: 1.5
```

```
11      max_allowed_time_to_collision_up_to_carrot: 1.0
12      max_angular_accel: 3.2
13      max_lookahead_dist: 0.9
14      max_robot_pose_search_dist: 1.4875
15      min_approach_linear_velocity: 0.05
16      min_lookahead_dist: 0.3
17      regulated_linear_scaling_min_radius: 0.9
18      regulated_linear_scaling_min_speed: 0.25
19      transform_tolerance: 0.1
20      use_cost_regulated_linear_velocity_scaling: true
21      use_interpolation: true
22      use_regulated_linear_velocity_scaling: true
23      use_rotate_to_heading: false
24      use_velocity_scaled_lookahead_dist: false
```

3. scripta: nav2_params.yaml

Kontrolagailu hau Ackermann mugimendu-sisteman oinarritzen diren robotentzat egokiena da, beraz, Summit-aren kasuan optimoa da. Konfigurazio horri esker, ibilbidearen jarraipen zehatza egitea lortzen da, talkak saihestuz eta ezarritako abiadura eta mugimendu mugei jarraituz. Robota atzeraka mugitu ahal izateko, kodeko 3. lerroan aurkitzen den parametroa “true” bezala egon behar da. Gainontzeko konfigurazioen informazioa, Nav2 webguneko dokumentazio ofizialean aurki daiteke [5].

BIBLIOGRAFIA

[1] ROS 2 Humble Hawksbill bertsioaren dokumentazio eta gida ofizialak:

https://docs.ros.org/en/humble/index.html

[2] GitHub-eko Summit-aren gordailua

https://github.com/lukin1225/inaki_ws.git

[3] Summit-ari buruzko informazioa eta 1. irudiaren web orria:
https://es.scribd.com/document/203263677/2-SUMMIT-System-Installation-and-Configuration-Manual

[4] The Construct plataforma:

https://www.theconstructsim.com/

[5] Nav2 liburutegiko dokumentazio ofiziala:

https://navigation.ros.org/