Ako naprogramovať myš v bludisku v ROSe - SK

Cieľom tohto návodu je ukázať ako riadiť robot simulovaný v Gazebo pomocou predpripravených ROS balíčkov.
Bude popísané ako spustiť simuláciu, nakonfigurovať bludisko a ako vytvoriť riadiaci algoritmus v C++ a Python.

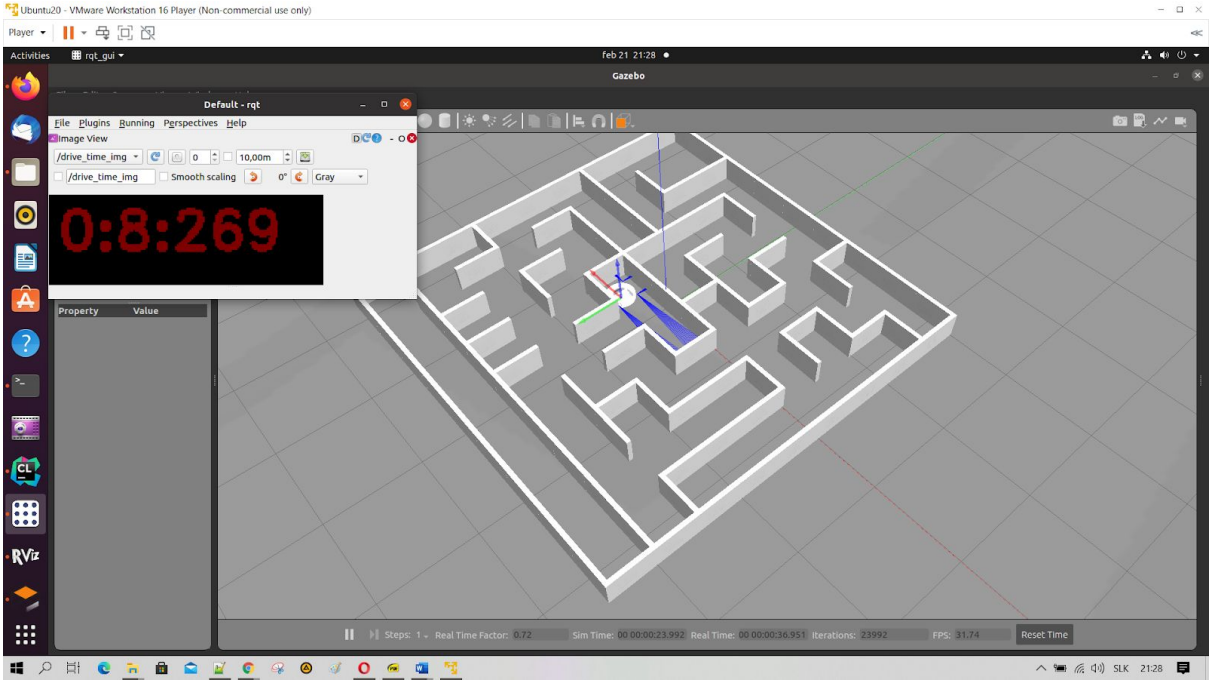
Spustenie

Predpokladom je mať nainštalovaný operačný systém Ubuntu 20 a ROS Noetic full desktop.
Potom je možné do vytvoreného a inicializovaného catkin_ws naklonovať alebo stiahnuť do src priečinka z repozitárov balíčky:
https://github.com/Smadas/irobot_create_ros - model iRobot Create
https://github.com/Smadas/maze_generator_ros - generovanie modelu bludiska z obrázka
https://github.com/Smadas/create_control_ros - riadiaci algoritmus pre simulovaný robot
Ďalej je potrebné stiahnuté zdrojové kódy skompilovať spustením konzolového príkazu catkin_make v umiestnení catkin_ws.
Následne je možné spustiť jeden zo spúšťacích súborov cez bash konzolu podľa toho či chceme spustiť iba simuláciu, riadenie alebo oboje a či riadenie má byť c++ alebo python:

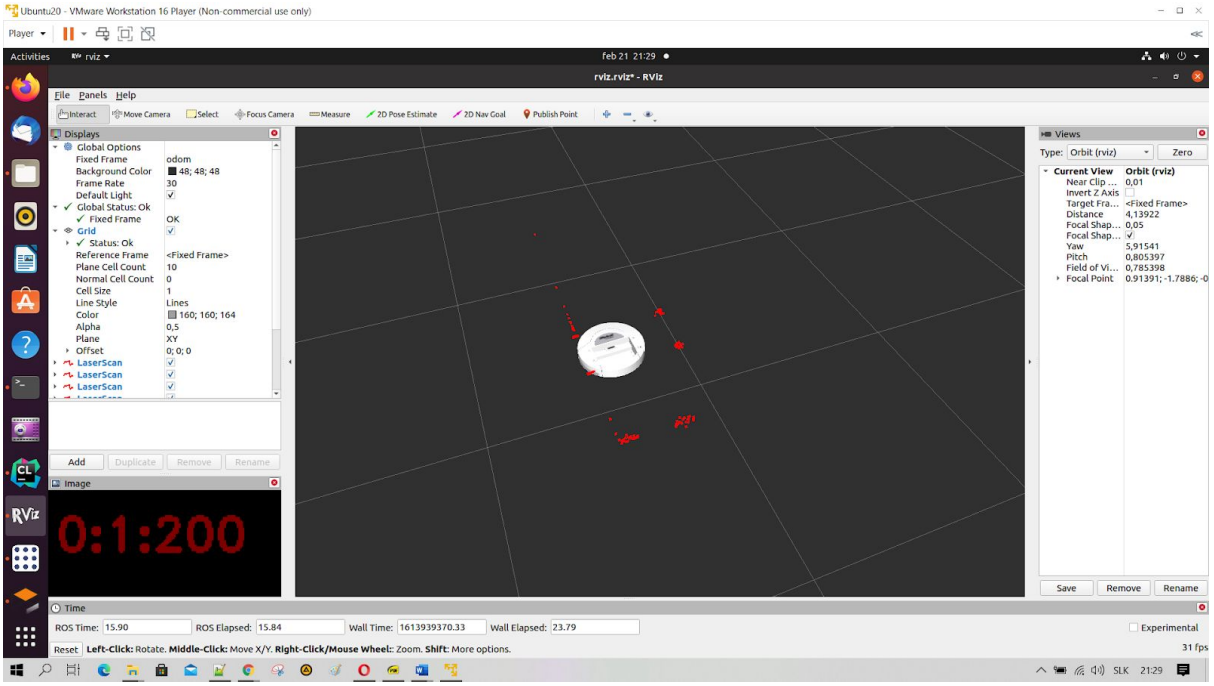
- control.launch
- sim_control.launch
- sim.launch
- control.launch
- sim_control.launch
- sim.launch

Spúšťací súbor sa spustí konzolovým príkazom:
roslaunch create_control <nazov_spustacieho_suboru>

Všetky procesy možno vypnúť stlačením ctrl+c v konzole cez ktorú sme ich spustili.



Simulácia Gazebo s bludiskom, robotom a časomierou.



Vizualizácia Rviz s robotom, diaľkomermi a časomierou.
V prípade, že bočný panel “zblbne”, stačí ho trošku roztiahnuť a už bude fungovať normálne.

Čo sa spustí

1. Generátor bludiska, vytvorí urdf a sdf model bludiska z .png obrázka bludiska. Ďalej vypočíta súradnice štartovacej a cieľovej plochy a štartovacej pozície robota.
2. Načítajú sa parametre do parameter servera.
3. Simulácia gazebo s vygenerovaným bludiskom a iRobot Create na vypočítanom mieste.
4. Riadiaci algoritmus (c++ alebo python).

5. Rviz - vizualizácia robota a všetkých senzorov, časomiera.
6. Časomiera - samostatné GUI zobrazujúce čas od kedy robot opustil ťažiskom štartovaciu plochu (ak sa zobrazuje iba prázdne okno treba nastaviť správny plugin na zobrazovanie z obrázku z menu>plugin>visualization>Image view a do textbox treba napísať /drive_time_img), pokým sa ťažiskom nedostane na cieľovú plochu. Vždy po prechode na cieľovú plochu sa čas zaznamená do súboru create_control/results/results.txt
7.

Súbor results.txt sa prepíše po novom spustení simulácie.

Upozornenie: Pri zmene bludiska je potrebné spusti a znovu vypnúť spustiteľný súbor aby sa zmeny prejavili.

Popis simulovaného robota

Simulovaný robot je modelom reálneho robota iRobot create s diferenciálnym podvozkom, kde jeho parametre sú:

- rozchod kolies 0.26m
- priemer kolies 0.066m

Simulovaný robot disponuje tromi typmi senzorov:

- IMU jednotkou - udáva otočenie okolo osí x, y, z - vyjadrené v radiánoch
- N-kódermi - udávajúcimi polohu kolies v radiánoch
- Diaľkomermi udávajúcimi vzdialenosť od prekážok
    - projekčný kužel 0.2rad
    - rozsahom 0.02m až 1m
    - s chybou s normálnym rozdelením 0.01m

Podrobné parametre simulovaného robota si možno pozrieť v súboroch mode-1_4.urdf a .sdf

Tvorba nového bludiska

Spustením launch súbor spúšťajúceho simuláciu sa spustí aj generovanie bludiska. Bludisko sa generuje z png súbora v umiestnení maze/maze_desc/ v ktorom sa nachádzajú predlohy bludiska.
Nové predlohy je možné nakresliť napríklad pomocou nástroja pikoPixel, stačí sa riadiť predlohou v priečinku maze_desc.
Po vytvorení predlohy je potrebné upraviť konfiguračný súbor maze/config/maze_config.yaml, kde sa nastavujú parametre stien, štartu a cieľa.
Nakoniec generator_maze vytvorí nový model bludiska a súbor obsahujúci súradnice štartovnej, cieľovej plochy a štartovnej pozície robota. Tieto súradnice sa berú do úvahy pri časomiere.

Programovanie riadiaceho algoritmu

Je možné si zvoliť programovací jazyk c++ (create_control/src/simple_control.cpp) alebo python (create_control/scripts/simple_control_py.py).

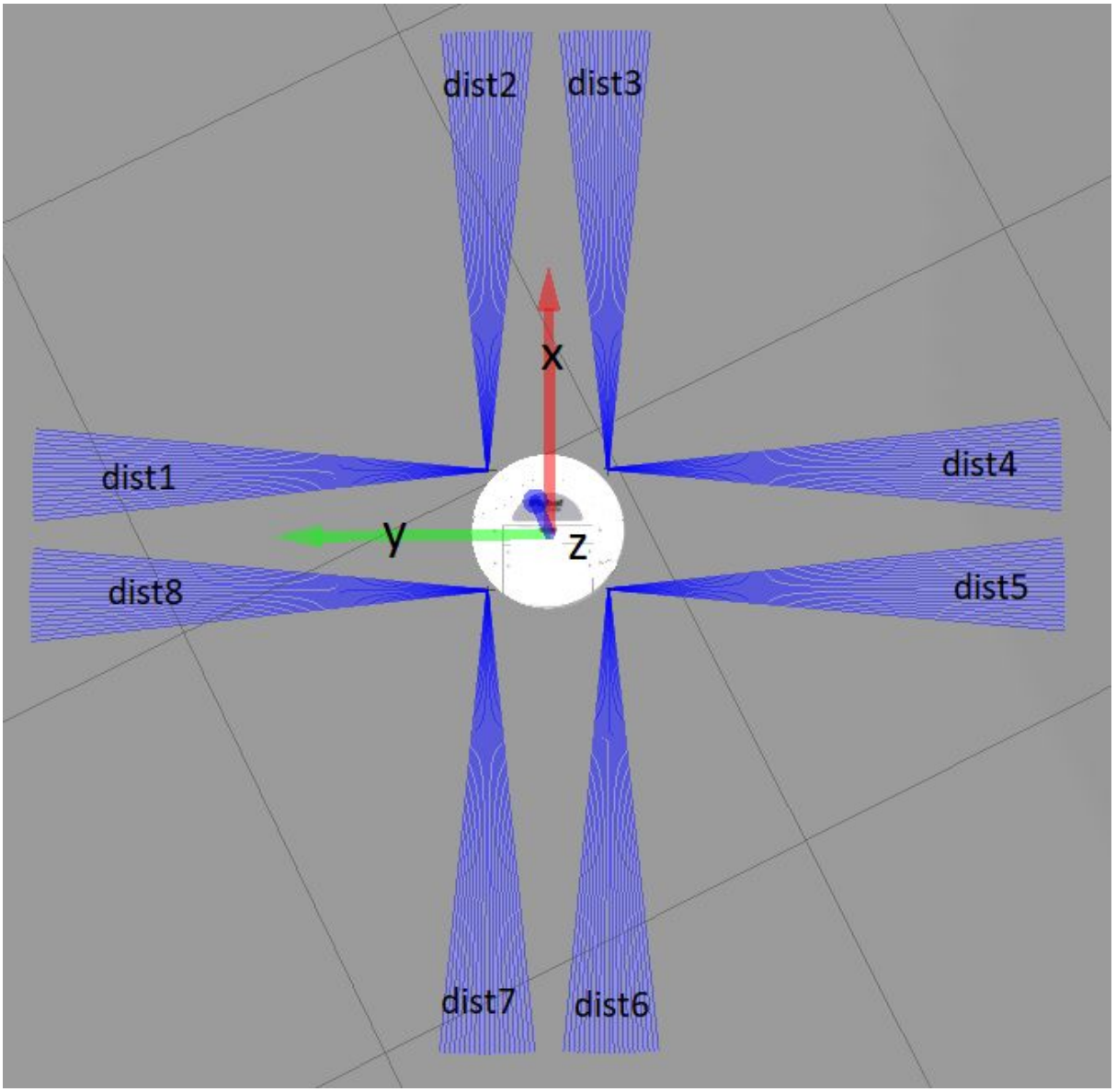
Dané súbory obsahujú komentármi vysvetlený zdrojový kód. Časti kódu, kde treba niečo doplniť sú označené komentárom //* popis *// pre c++ a
##* popis *## pre python. Samozrejme je možné pridávať aj ďalšie súbory a upravovať ostatné, avšak pre jednoduchosť je algoritmus vhodné tvoriť v predznačených miestach. Pred spustením python riadiaceho algoritmu ho treba nastaviť na spúšťanie pomocou nasledovného príkazu v jeho umiestnení:
chmod +x create_control_py.py
Nezabudnúť vždy po úprave zdrojového kódu spustiť kompiláciu konzolovým príkazom catkin_make v umiestnení catkin_ws.
Robot sa riadi nekonečnou slučkou s pravidelnou frekvenciou čítania hodnôt zo senzorov a zapisovania rýchlostí do robota.

Riadiaca správa

Robot je riadený správou typu Twist, ktorá obsahuje rotačnú rýchlosť v osiach x, y, z a primočiaru rýchlosť v osiach x, y, z, výsledný pohyb robota je vektorovým súčinom týchto rýchlostí.
geometry_msgs::Twist cmd_vel
cmd_vel.angular.x = 0
cmd_vel.angular.y = 0
cmd_vel.angular.z = 0
cmd_vel.linear.x = 0
cmd_vel.linear.y = 0
cmd_vel.linear.z = 0

Dáta zo senzorov

Robot publikuje správu typu create_sensors, ktorá obsahuje vzdialenosti merané diaľkomermi, rotáciu robota simulovanou IMU jednotkou(kompas) a hodnoty n-kóderov určujúcich polohu kolies.
create_control::create_sensors sensors
sensors.dist1 = 0 - (metre) priemerná vzdialenosť od prekážky average distance of obstacle (simulácia ultrazvukového diaľkomera)
sensors.dist2 = 0
sensors.dist3 = 0
sensors.dist4 = 0
sensors.dist5 = 0
sensors.dist6= 0
sensors.dist7 = 0
sensors.dist8 = 0
sensors.imu_p = 0 - uhol okolo osi x (rad)
sensors.imu_r = 0 - uhol okolo osi y (rad)
sensors.imu_y = 0 - uhol okolo osi z (rad)
sensors.left_wheel = 0 - uhol otočenia ľavého kolesa (rad)
sensors.right_wheel = 0 - uhol otočenia pravého kolesa (rad)



Pre pokročilých vývojárov

Príklady riadiacich algoritmov sú realizované najjednoduchším možným spôsobom aby mohli balíčky použiť aj začiatočníci, avšak je možné použiť aj ostatné dostupné nástroje napríklad prerobiť základný c++, python node na triedu.
Ďalej je možné použiť ďalšie knižnice a ros balíčky. V tomto prípade však treba upraviť súbor CmakeLists.txt a package.xml v príslušnom balíčku. Konkrétne je potrebné pridať:

- do CmakeLists.txt find_package(....) príslušný pridaný balíček
- do CmakeLists.txt include_directories(.....) cestu k príslušným hlavičkovým súborom
- do package.xml dependencies na daný balíček (možno odpozorovať od iných balíčkov)

Ďalej je možné vytvoriť si vlastné ROS topic alebo service na prijímanie alebo odosielanie údajov. Návody môžete nájsť na: http://wiki.ros.org/ROS/Tutorials