**SIMULAÇÃO:**

Iremos simular o roubo Turtlebot 3 "Burger", um modelo de robô real desenvolvido

para estudos.

O simulador utilizado é o gazebo, incluído no ROS-FULL-DESKTOP.

O fabricante do robô usado, fornece inúmeros pacotes pré configurados que podem ser usados com a simulação e que vão ajudar no desenvolvimento do nosso trabalho.

**Instalando a SIMULAÇÃO:** (<http://dainf.ct.utfpr.edu.br/~andre/lib/exe/fetch.php?media=robmov_aula4_experimentacao_virtual.pdf>):

sudo apt-get update

sudo apt-get install ros-kinetic-turtlebot3

cd ~/catkin\_ws/src/

git clone https://github.com/ROBOTIS-GIT/turtlebot3\_simulations.git

cd ~/catkin\_ws && catkin\_make

**Executando a Simulação:**

Preparando o terminal:

cd ~/catkin\_ws

source devel/setup.bash

Selecionar uma das versões:

export TURTLEBOT3\_MODEL=burger << Usamos Essa!

export TURTLEBOT3\_MODEL=waffle$

Executando de fato o programa:

roslaunch turtlebot3\_gazebo turtlebot3\_house.launch

Caso o robô não apareça no mapa da simulação:

sudo rosdep init

rosdep update

ou

rosdep update

Lembre-se:

Se tudo deu certo já é possível utilizar o rqt para monitorar e controlar o robo!

**SLAM**

O SLAM é um algoritmo que permite que o robô o ambiente físico ao redor enquanto se locomove por ele. Através de sensores e da própria odometria, o robô é capaz de criar mapas, ou seja, ele aprende mais sobre os obstáculos em torno dele a cada vez que ele

passa por eles.

O Turtlebot 3 possui 2 tipos de mapeamento, fornecido pelo fabricante:

* Gmapping : É o que vamos usar.
* Cartographer: Não consegui fazer funcionar.

\*/

**Instalando o SLAM:**

Aqui são instalados os pacotes capazes de reconhecer os componentes do robô como sensores e etc, além de baixar também as estratégias de navegação. (http://emanual.robotis.com/docs/en/platform/turtlebot3/slam/#slam):

sudo apt-get update

sudo apt-get install ros-kinetic-joy ros-kinetic-teleop-twist-joy ros-kinetic-teleop-twist-keyboard ros-kinetic-laser-proc ros-kinetic-rgbd-launch ros-kinetic-depthimage-to-laserscan ros-kinetic-rosserial-arduino ros-kinetic-rosserial-python ros-kinetic-rosserial-server ros-kinetic-rosserial-client ros-kinetic-rosserial-msgs ros-kinetic-amcl ros-kinetic-map-server ros-kinetic-move-base ros-kinetic-urdf ros-kinetic-xacro ros-kinetic-compressed-image-transport ros-kinetic-rqt-image-view ros-kinetic-gmapping ros-kinetic-navigation ros-kinetic-interactive-markers

sudo apt-get install ros-kinetic-cartographer ros-kinetic-cartographer-ros ros-kinetic-cartographer-ros-msgs ros-kinetic-cartographer-rviz

sudo apt-get install ros-kinetic-hector-mapping

sudo apt-get install ros-kinetic-slam-karto

sudo apt-get install ros-kinetic-frontier-exploration ros-kinetic-navigation-stage

Instalando o Cartographer (Creio que seja opcional, pois vamos usar o gmapping):

sudo apt-get install ninja-build libceres-dev libprotobuf-dev protobuf-compiler libprotoc-dev

cd ~/catkin\_ws/src

git clone https://github.com/googlecartographer/cartographer.git

git clone https://github.com/googlecartographer/cartographer\_ros.git

cd ~/catkin\_ws

src/cartographer/scripts/install\_proto3.sh

rm -rf protobuf/

rosdep install --from-paths src --ignore-src -r -y --os=ubuntu:xenial

catkin\_make\_isolated --install --use-ninja

**Executando o mapeamento:**

Primeiro, execute a simulação, descrito na seção anterior. Após este passo finalizado, abra um novo terminal, que será usado para executar o SLAM.

Os dois passos abaixo são para o "Cartographer", como não consegui fazer funcionar

esqueça por enquanto!

source ~/catkin\_ws/install\_isolated/setup.bash

roslaunch turtlebot3\_slam turtlebot3\_slam.launch slam\_methods:=cartographer

Os dois passos abaixo são para rodar o Gmapping, técnica que usaremos!

Selecionar uma das versões:

export TURTLEBOT3\_MODEL=burger << Usamos Essa!

export TURTLEBOT3\_MODEL=waffle$

roslaunch turtlebot3\_slam turtlebot3\_slam.launch slam\_methods:=gmapping

A partir de agora haverá uma tela do programa RVIZ exibindo o mapa gerado a partir das informações que o robô colheu do ambiente simulado!

Você deve fazer robô aprender a casa inteira!

Para salvar o mapa em outra janela de terminal digite:

rosrun map\_server map\_saver -f ~/map

Feche o RVIZ:

**Navegação**

Nesta parte, parte-se do princípio que seu mapa está completo.

Assim, na mesma janela de terminal (assim vc não precisa digitar o export TURLEBOT3\_model = $) em que vc rodou o mapeamento, digite o seguinte comando:

Selecionar uma das versões:

export TURTLEBOT3\_MODEL=burger << Usamos Essa!

export TURTLEBOT3\_MODEL=waffle$

roslaunch turtlebot3\_navigation turtlebot3\_navigation.launch map\_file:=$HOME/map.yaml

O RVIZ será executado, mas desta vez no modo “navegação”, então nesta janela use o botão “Set Current Position” para setar onde o robô ele está no gazebo (Ele não sabe onde começar) e uma vez que vc consiga coincidir o “Local Map” com o ponto de partida o sistema está pronto para navegar no mapa desenvolvido no passo anterior.

PS: Tentei rodar sem um mapa prévio, mas o Turtlebot tem uma odometria, muito mas muito ruim (Isso porque ele usa rodas para se locomover!) e causava uma aberração chamada “drift”.