SISTEMA BASEADO EM ROS DISTRIBUÍDO PARA CONTROLO DE UMA PLATAFORMA SKID-STEERING

Filipe Aguiar da Silva
Orientador: **Vítor Santos**
Coorientador: **Miguel Oliveira**

Mestrado Integrado em Engenharia Mecânica
Universidade de Aveiro
Aveiro, 14 de julho de 2017

Agenda

1. Enquadramento e Motivação
2. Objetivos
3. Infraestrutura experimental e ferramentas
4. Controlo da Plataforma
5. Sistema Computacional
6. Ambiente de Simulação
7. Uma Aplicação Ilustrativa
8. Conclusões e trabalho futuro

1. Enquadramento e Motivação

- Plataforma NARDO apresentada pela empresa Tarento Robotics
- Plataforma skid-steering
- Implementação de uma arquitetura ROS

2. Objetivos

- Criação de uma rede de unidades computacionais para a execução de ROS numa abordagem distribuída;
- Instalação do ROS e desenvolvimento do software necessário à interligação da plataforma com sistemas periféricos;
- Criação de um ambiente de simulação para uma plataforma skid-steering;
- Criação de uma aplicação de demonstração.

3. Infraestrutura experimental e ferramentas

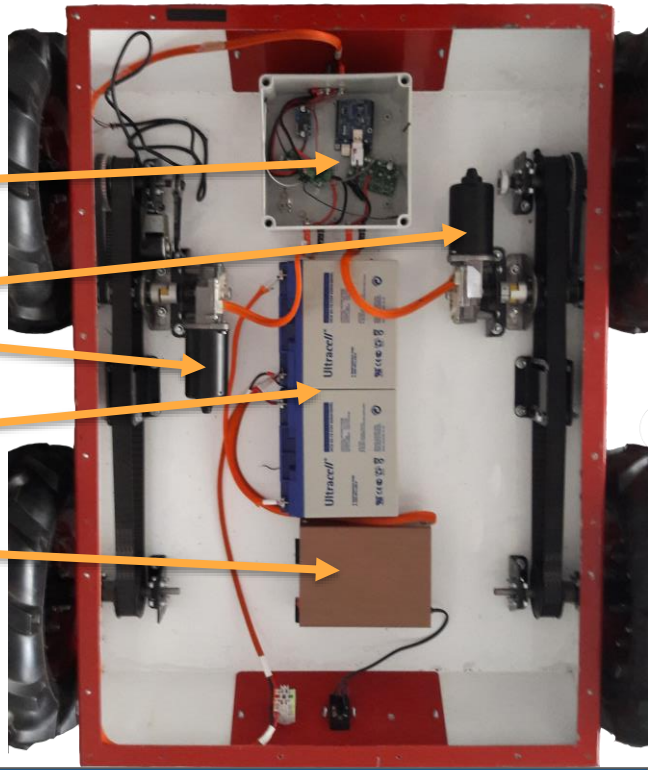
Plataforma Robótica Móvel

Drivers de potência e controlador

2 x motor DC 24V

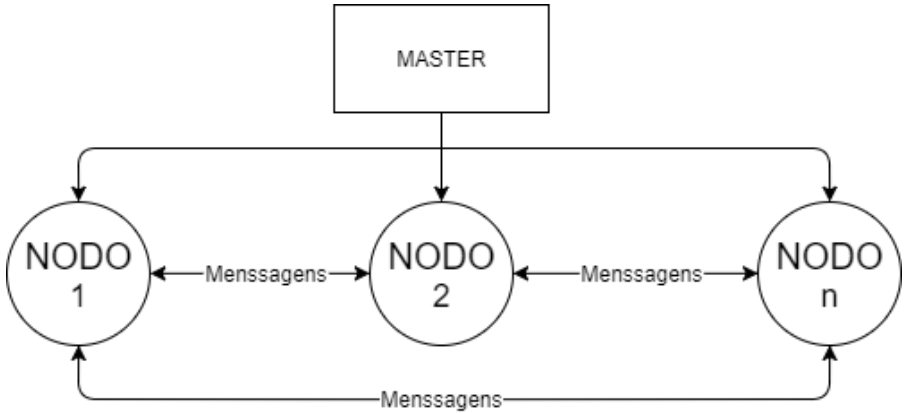
2 x 12V baterias em série

Carregador de barerias



ROS - Robotic Operating System

- Framework criada para o desenvolvimento de robôs
- Baseado na publicação e subscrição de mensagens



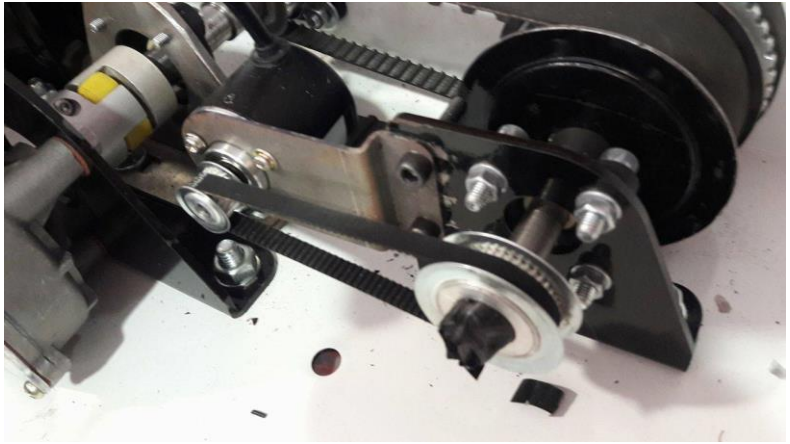
4. Controlo da plataforma

Controlo da velocidade

- Aplicação de um controlador PID
- Necessidade da instalação de encoders
- Necessidade de comunicação com a unidade computacional

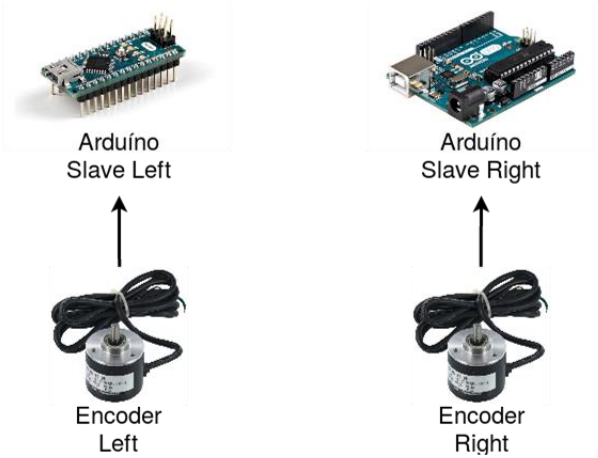
Instalação de encoders

Tensão mínima	5V
Tensão máxima	24V
Nº de canais	2
Rotação máxima	5000RPM
Pulsos por rotação	100
Diâmetro do veio	6mm
Diâmetro exterior	40mm



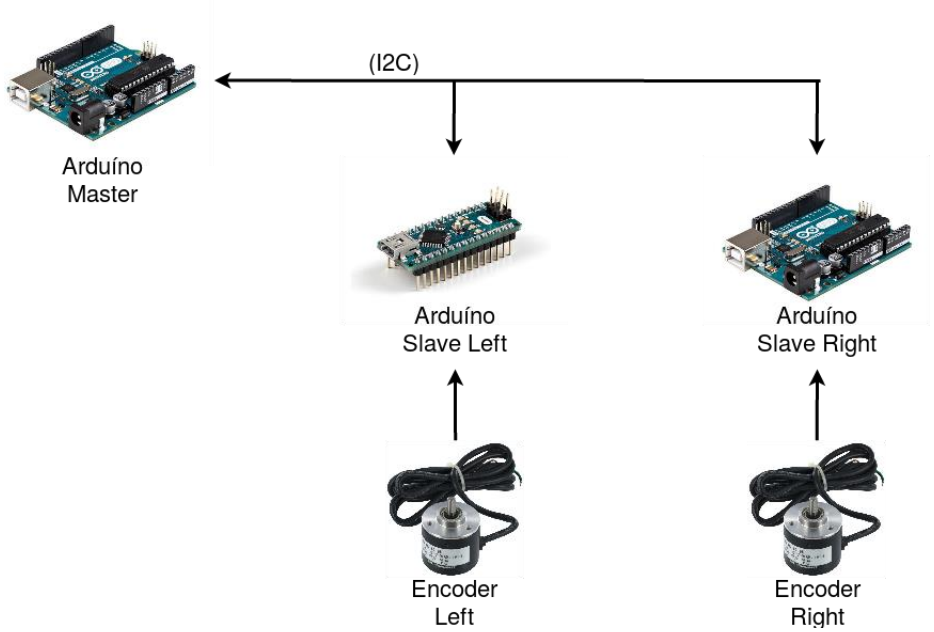
Leitura dos encoders

- Um controlador Arduino dedicado a cada encoder
- Uso dos dois sinais do encoder para a leitura



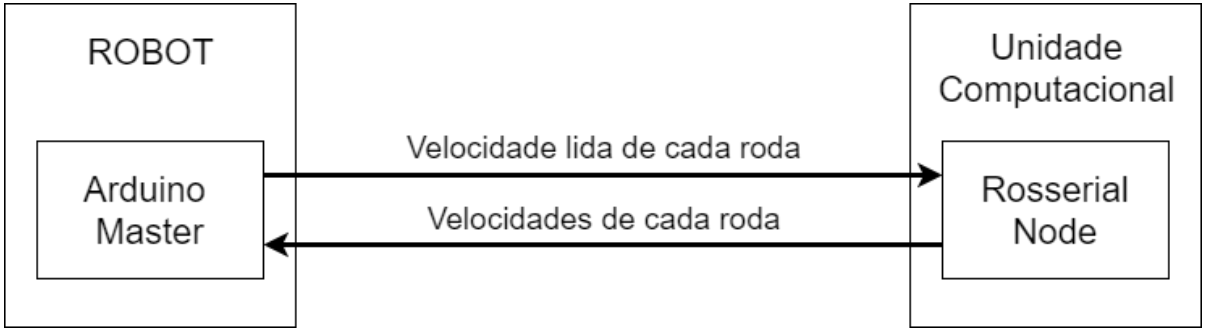
Leitura dos encoders

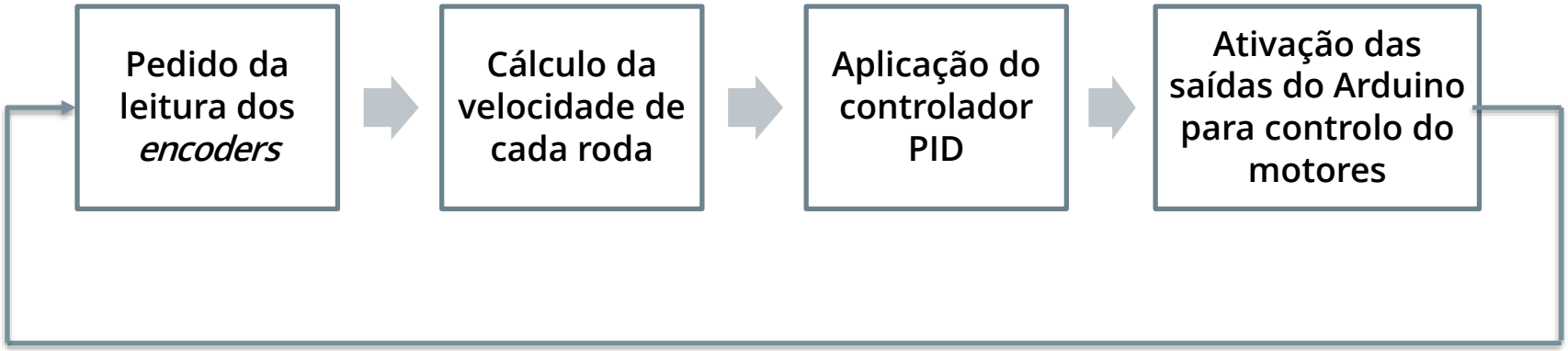
- Um controlador Arduino dedicado a cada encoder
- Uso dos dois sinais do encoder para a leitura
- Informação enviada para Arduino Master por I2C



Comunicação com a unidade computacional

- Utilização do package Rosserial
- Envio das velocidades desejadas de cada roda
- Receção das velocidades lidas




Pedido da leitura dos *encoders*
Cálculo da velocidade de cada roda
Aplicação do controlador PID
Ativação das saídas do Arduino para controlo do motores

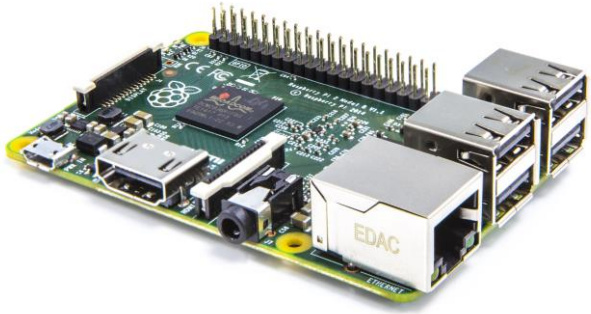
Limitações da plataforma

- Problema detetado num dos motores
- Continuação do trabalho recorrendo a simulações
- Impossibilidade de aquisição de uma unidade computacional dedicada
- Uso de unidade computacional de baixo custo

5. Sistema computacional

Raspberry Pi

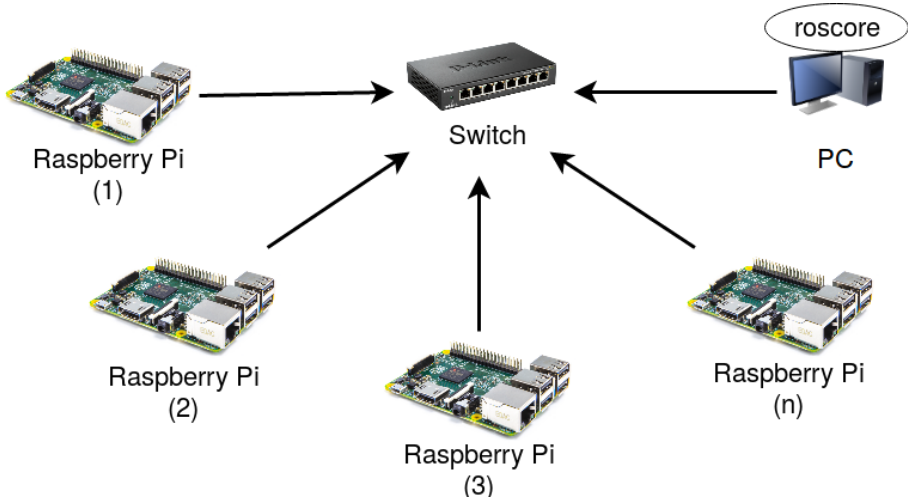
- Microprocessador
- Pequenas dimenções
- Mesmas funcionalidades de um computador



	Raspberry Pi 2	Raspberry Pi 3
Processador	Cortex-A7	Cortex-A53 64-bit
Nº de núcleos	4	4
CPU Clock	900MHz	1.2GHz
GPIO pinos	40	40
Porta HDMI	Sim	Sim
Porta Ethernet	Sim	Sim
Portas USB	4	4
Wi-Fi	Não	Sim
Bluetooth	Não	Sim

Solução proposta

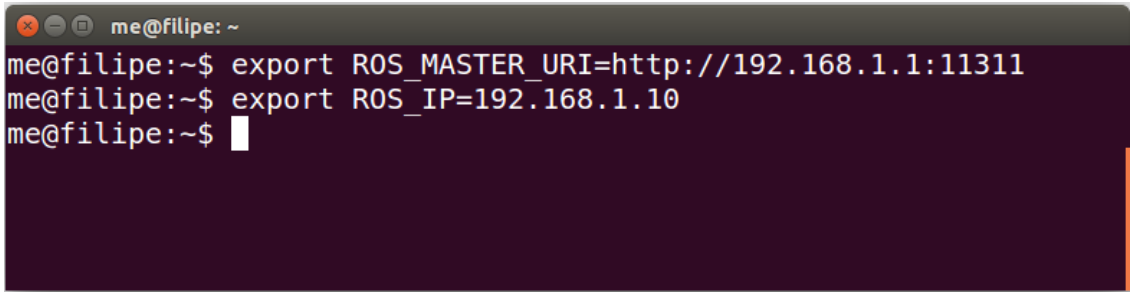
- Uso de diversos Raspberry Pi's
- Uma unidade computacional a correr o *roscore*
- Nodos ROS a serem processador nos Raspberry Pi's
- IP fixo a cada Raspberry Pi



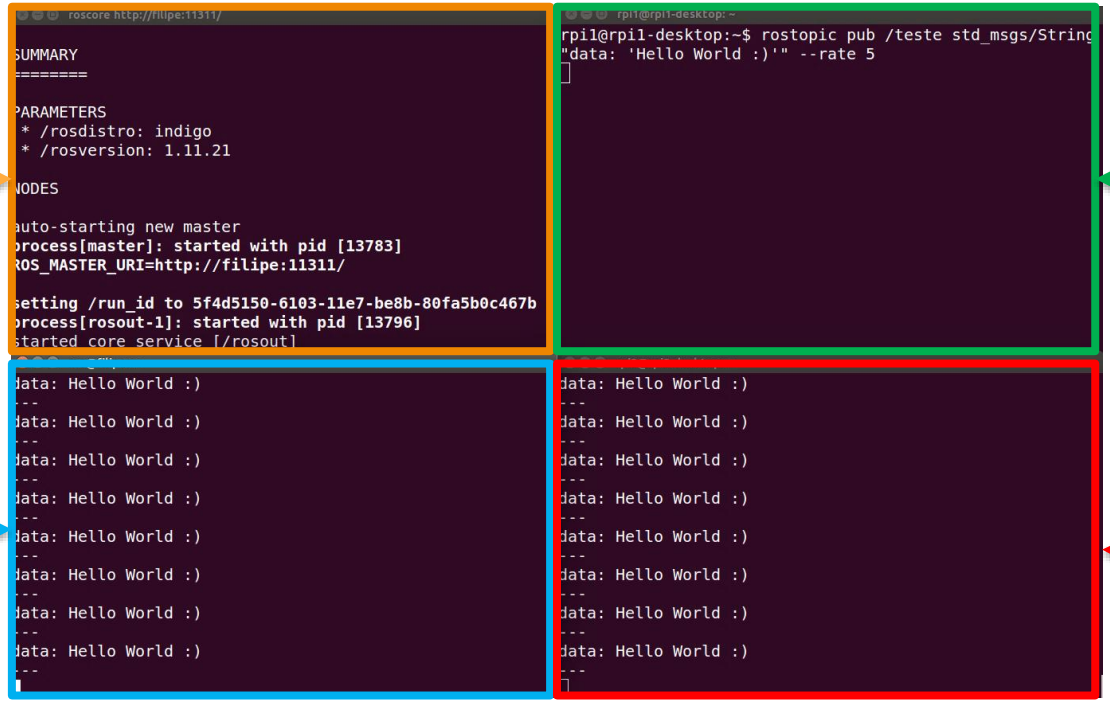
Procedimento

Configuração de cada unidade:

1. **ROS_MASTER_URI**
2. **ROS_IP**



ROS Master



Publicação de um tópico num Raspberry Pi

Visualização do tópico publicado no computador

Visualização do tópico publicado num Raspberry Pi

Ambiente de desenvolvimento

Sistema de partilha de ficheiros

- Pasta partilhada
- Network File System (NFS)

Controlo remoto

- SSH
- Inicialização simultânea

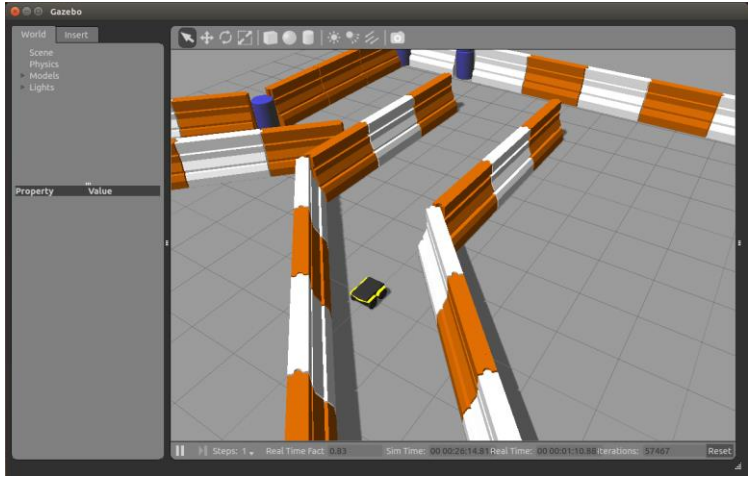
Launch file **gerais**

- Inicialização de vários nodos em diferentes máquinas simultaneamente

6. Ambiente de simulação

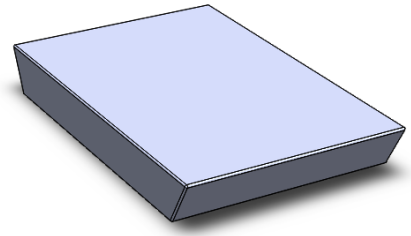
Gazebo

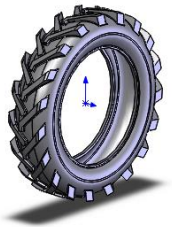
- Simulador 3D
- Integração com ROS
- Simulação tendo em conta propriedades fisícas dos materias (massa, fricção)
- Possibilidade de adicionar sensores de odometria, visão e distância



Definição do Robô

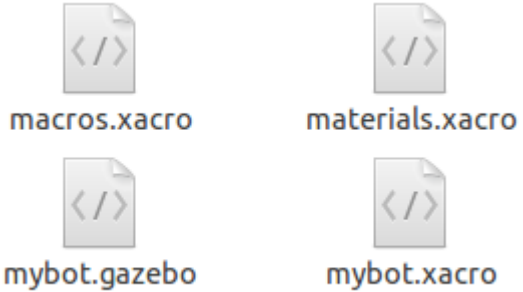
- Desenho dos componentes em software CAD





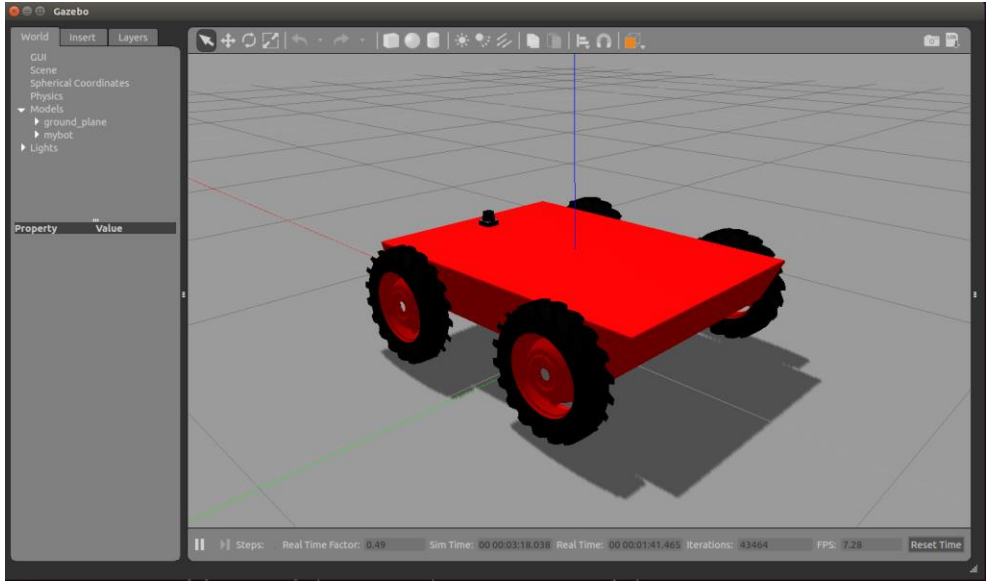
Definição do Robô

- Desenho dos componentes em software CAD
- Definição do modelo URDF



Vizualização no simulador

- Criação de *launch file* específica



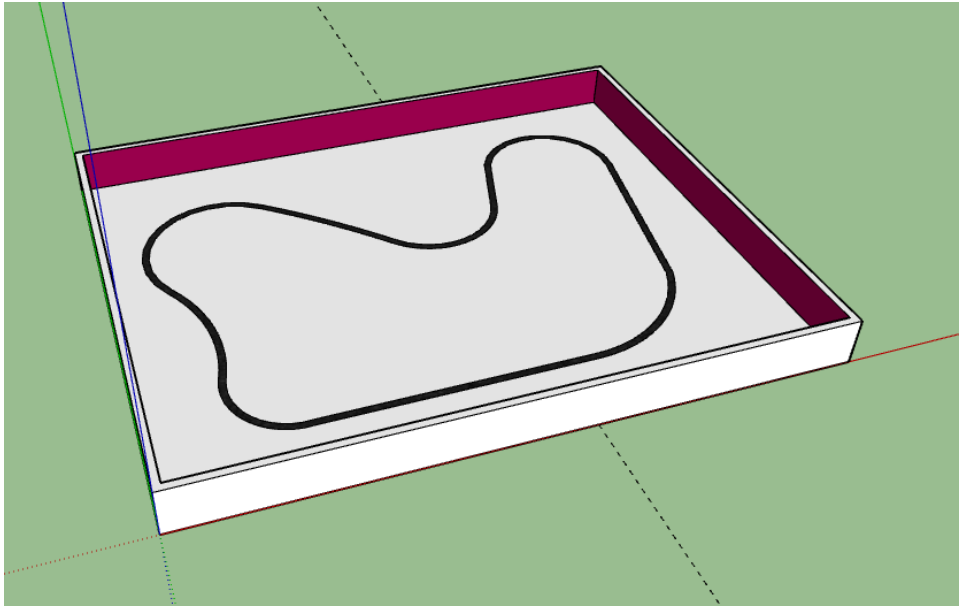
7. Uma aplicação ilustrativa

Proposta

- Utilização do simulador desenvolvido e sistema computacional
- Aplicação baseada no seguimento de uma linha
- Ter a opção de controlo da plataforma usando um *gamepad*

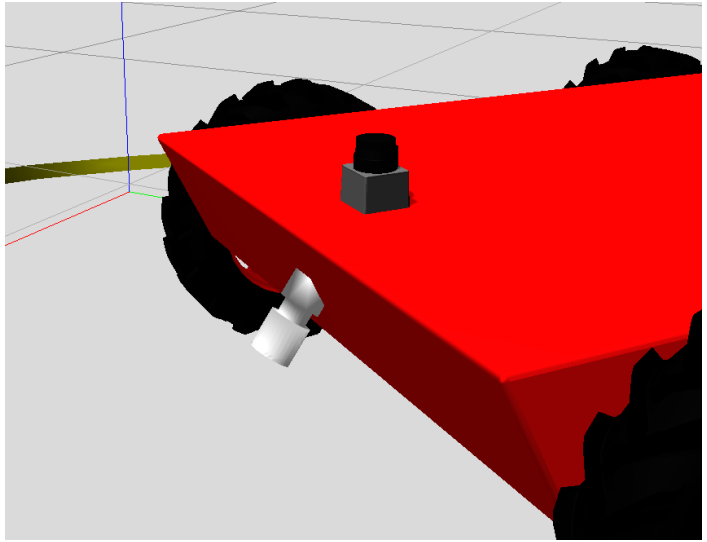
Desenvolvimento

- Criação da pista



Desenvolvimento

- Criação da pista
- Implementação dos sensores



Desenvolvimento

- Criação da pista
- Implementação dos sensores
- Criação de quatro nodos ROS

Line Node

- Seguimento da linha
- Alinhamento da plataforma quando se encontra perdida

Laser Node

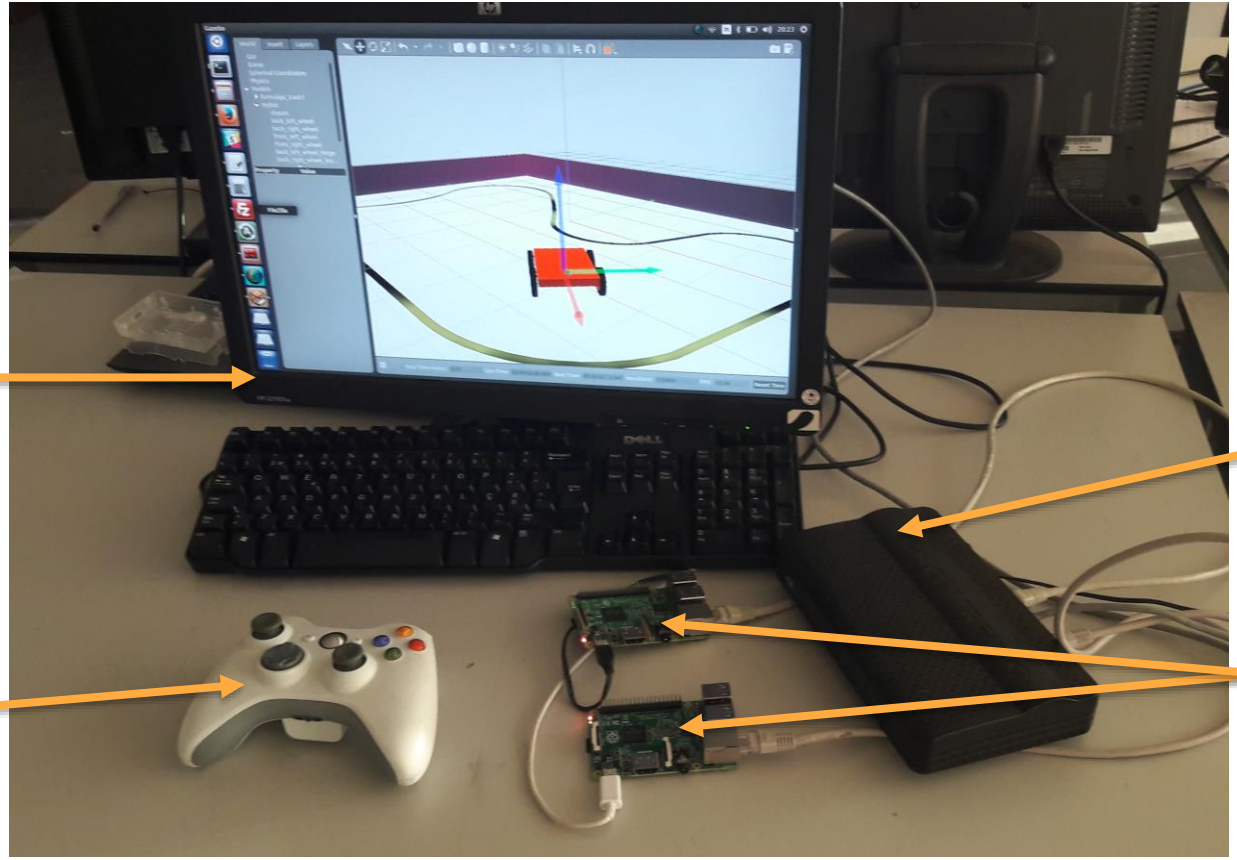
- Evitar que plataforma passe os limites

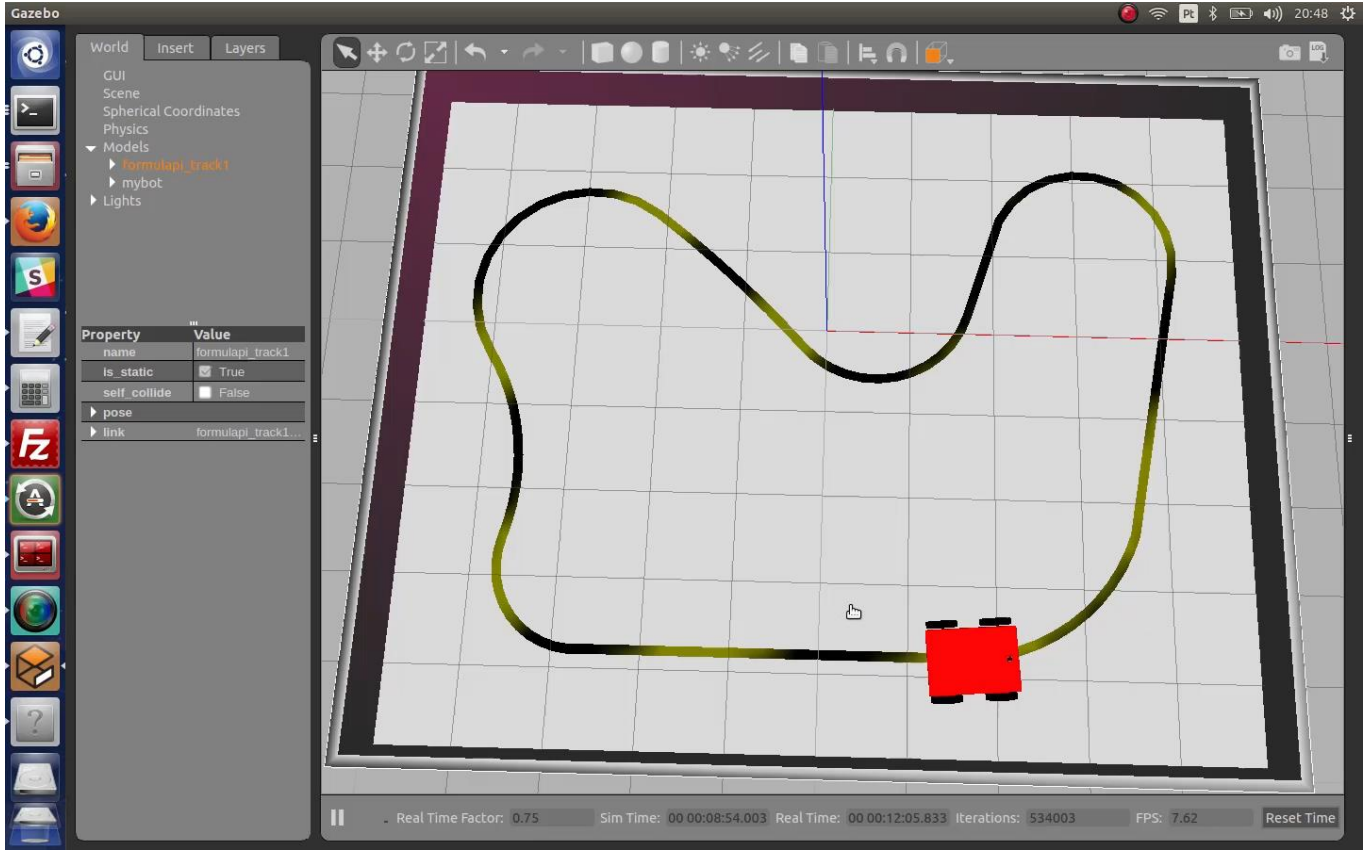
Joy Node

- Controlo da plataforma usando um *gamepad*

Decision Node

- Decide qual dos nodos controla a plataforma
- Laser node > Joy node > Line node


Computador
Switch
Gamepad
Raspberry Pi's


Gazebo
20:48
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
formulapi_track1
mybot
Lights
Property
Value
name
formulapi_track1
is_static
True
self_collide
False
pose
link
formulapi_track1...
Real Time Factor: 0.75
Sim Time: 00 00:08:54.003
Real Time: 00 00:12:05.833
Iterations: 534003
FPS: 7.62
Reset Time

8. Conclusões e trabalho futuro

Conclusões

- A solução encontrada para o controlo da plataforma foi adequada
- Foi criada a rede computacional utilizando ROS numa abordagem distribuída
- O ambiente de simulação permitiu fazer a simulação de uma plataforma skid-steering
- A aplicação desenvolvida permitiu o teste do sistema computacional e simulador

Trabalho Futuro

- Compilação cruzada
- Teste da solução em uma plataforma real
- Continuação do trabalho feito na plataforma

OBRIGADO.

SISTEMA BASEADO EM ROS DISTRIBUÍDO PARA CONTROLO DE UMA PLATAFORMA SKID-STEERING

Filipe Aguiar da Silva
Orientador: **Vítor Santos**
Coorientador: **Miguel Oliveira**

Mestrado Integrado em Engenharia Mecânica
Universidade de Aveiro
Aveiro, 14 de julho de 2017