ROS Workshop

Miguel Oliveira, Héber Sobreira

April 21, 2015

CROB - Centro de Robótica e Sistemas Inteligentes
INESC TEC

Table of contents

Objectives

- Lets play a game called **Team Hunt** (just made that up, don't bother googling)
- Basic rules:
    - Three or more teams of one or more players
    - Players on a team hunt players from another team, while at the same time they must evade players from a different other team
        - Ex. *Team A* hunts *Team B* hunts *Team C* hunts *Team A*
    - A player is killed if another player (from a team hunting him) gets close enough (bellow a threshold)
    - The team which has hunted more players wins the game
- The game takes place in an arena with fixed dimensions (in 2D)
- Each player will have a turn to decide how to move.

What we will learn to do in ROS

- Creating ROS packages, nodes and libraries, compiling and running them
- Using ROS based communications, publish / subscribe, server / response
- Using ROS launch file scripts to ease the startup of complex systems
- Using ros parameters to configure the nodes
- Creating custom messages
- Using RVIZ and publishing visualization markers
- Visualizing the ROS nodes and the topics they are exchanging
- Using the ROS tf library (just the basics)
- Using the rosbag tool to record the system's output
- a lot more I don't remember right now . . .

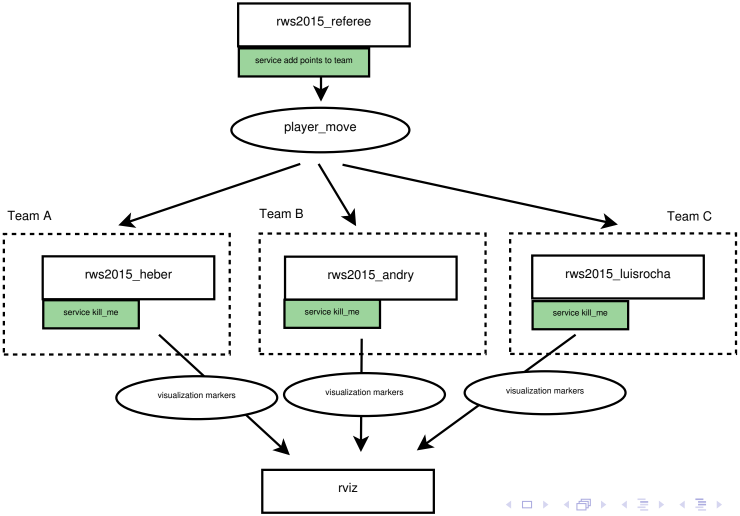
What must be done before starting the workshop

- Install Ubuntu 14.04 64bit and ROS indigo
  - Other configurations are possible, but if something goes wrong we will not have time to try to solve the problems
- Tutorials 1 through 16 from `http://wiki.ros.org/ROS/Tutorials` should be completed before starting the workshop
  - use the catkin (and not rosbuild) to build your packages
  - you can skip the python tutorials
- C++ tutorials at `http://www.cplusplus.com/doc/`, take a look if you have no C++ background

Software architecture

- The game runs in multiple processes, which communicate with each other
- Each player is a process (a ros node), which will have the following interfaces:
    - Subscribe to a message (*player_move*) which will give the command to a player to move
    - Broadcast a tf transform to let other players know about its position
    - Provide a service (*kill_me*), to be called by the hunters
    - Access a service (*add_points_to_team*) provided by the referee
    - Publish rviz markers to show his position in the arena, along with some *nasty comments* to intimidate the adversaries

Example


rws2015_referee
service add points to team
player_move
Team A
Team B
Team C
rws2015_heber
service kill_me
rws2015_andry
service kill_me
rws2015_luisrocha
service kill_me
visualization markers
visualization markers
visualization markers
rviz