ROS

Akka Ecole Technology

Overview

Today's Goals

- Overview of ROS

Today's Goals

- Overview of ROS
- Using built-in ROS tools

Today's Goals

- Overview of ROS
- Using built-in ROS tools
- Writing custom ROS nodes

Today's Overview

- 9:00 What is ROS and why do we want it

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:45 Start with the ROS tutorials

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break
- 13:30 Continue with the ROS Tutorials

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break
- 13:30 Continue with the ROS Tutorials
- 14:45 Coffee Break

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break
- 13:30 Continue with the ROS Tutorials
- 14:45 Coffee Break
- 15:00 Build a ROS node which uses deep learning to process MNIST images

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break
- 13:30 Continue with the ROS Tutorials
- 14:45 Coffee Break
- 15:00 Build a ROS node which uses deep learning to process MNIST images
- 16:00 Small ROS Quiz: Use the ROS tools on a Rosbag from the RC-Car

Today's Overview

- 9:00 What is ROS and why do we want it
- 10:30 Coffe Break
- 10:45 Start with the ROS tutorials
- 12:30 Lunch Break
- 13:30 Continue with the ROS Tutorials
- 14:45 Coffee Break
- 15:00 Build a ROS node which uses deep learning to process MNIST images
- 16:00 Small ROS Quiz: Use the ROS tools on a Rosbag from the RC-Car
- 17:00 Checking who Survived :P

Introduction to ROS

Building a Robot

- Without ROS

Building a Robot

- Without ROS
  - Build device drivers

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...
  - Implement logging, control and error handling

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...
  - Implement logging, control and error handling
- With ROS

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...
  - Implement logging, control and error handling
- With ROS
  - Logging, error handling, communications framework, driver for standard devices and even some perception, navigation and motion planning algorithms are already provided.

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...
  - Implement logging, control and error handling
- With ROS
  - Logging, error handling, communications framework, driver for standard devices and even some perception, navigation and motion planning algorithms are already provided.
  - There is a package management system included, making the development way easier.

Building a Robot

- Without ROS
  - Build device drivers
  - Build a communications framework
  - Write algorithms for perception, navigation, motion planning,...
  - Implement logging, control and error handling
- With ROS
  - Logging, error handling, communications framework, driver for standard devices and even some perception, navigation and motion planning algorithms are already provided.
  - There is a package management system included, making the development way easier.
  - It provides tools for visualisation, simulation and analysis

History of ROS

- Development started in 2000 in the Stanford Artificial Intelligence Lab as project Switchyard.

History of ROS

- Development started in 2000 in the Stanford Artificial Intelligence Lab as project Switchyard.
- In 2007 it became a formal entity with support from Willow Garage.

History of ROS

- Development started in 2000 in the Stanford Artificial Intelligence Lab as project Switchyard.
- In 2007 it became a formal entity with support from Willow Garage.
- Since 2013 the Open Source Robotics Foundation is maintaining and developing ROS.

History of ROS

- Development started in 2000 in the Stanford Artificial Intelligence Lab as project Switchyard.
- In 2007 it became a formal entity with support from Willow Garage.
- Since 2013 the Open Source Robotics Foundation is maintaining and developing ROS.
- The primary motivation behind ROS was the recognition that researchers were constantly reinventing the wheel as there were no reusable components for robotics projects to start from.

History of ROS

- Development started in 2000 in the Stanford Artificial Intelligence Lab as project Switchyard.
- In 2007 it became a formal entity with support from Willow Garage.
- Since 2013 the Open Source Robotics Foundation is maintaining and developing ROS.
- The primary motivation behind ROS was the recognition that researchers were constantly reinventing the wheel as there were no reusable components for robotics projects to start from.
- In addition, it was hard for researchers to exchange and validate code from other researchers.

High Level Robot Tasks

High Level Robot Tasks

High Level Robot Tasks

Perception

High Level Robot Tasks

Perception

Decision Making

High Level Robot Tasks

Perception

Decision Making

Actuation

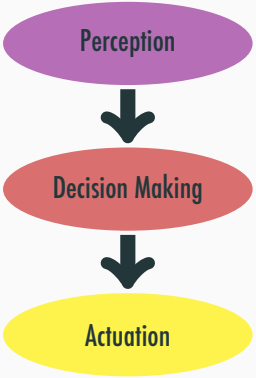
High Level Robot Tasks

Perception

Decision Making

Actuation

High Level Robot Tasks



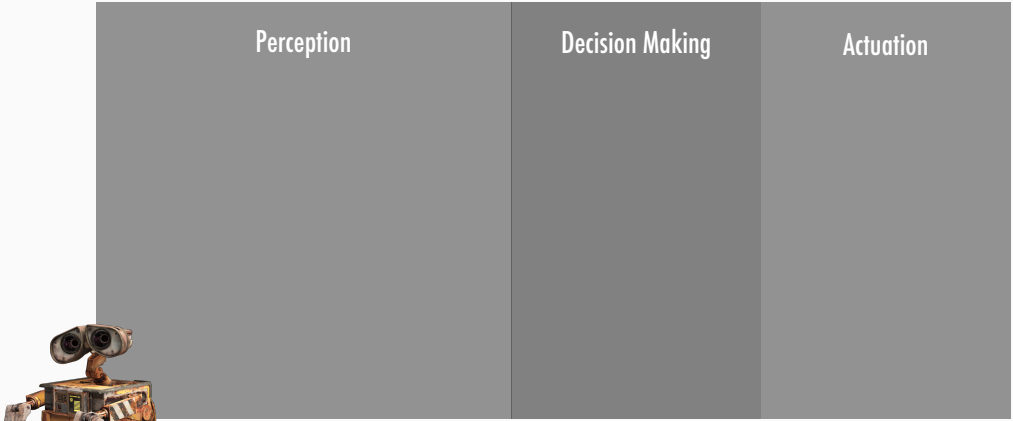
Literally every robot is performing these high level tasks.

Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.

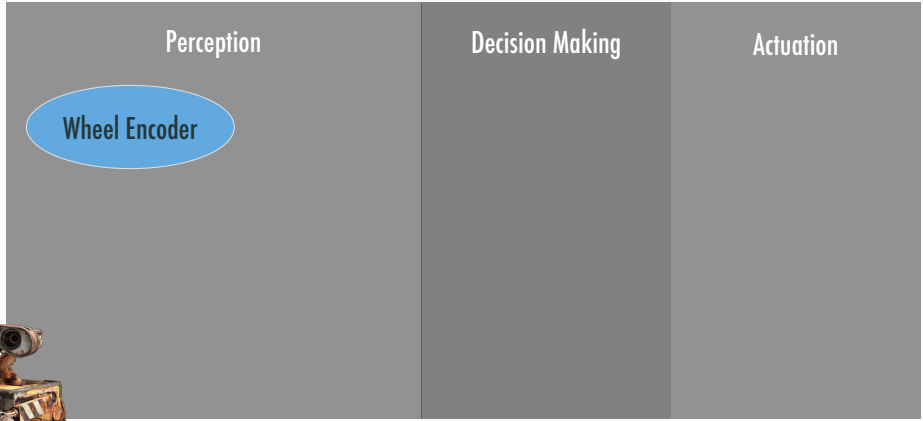
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



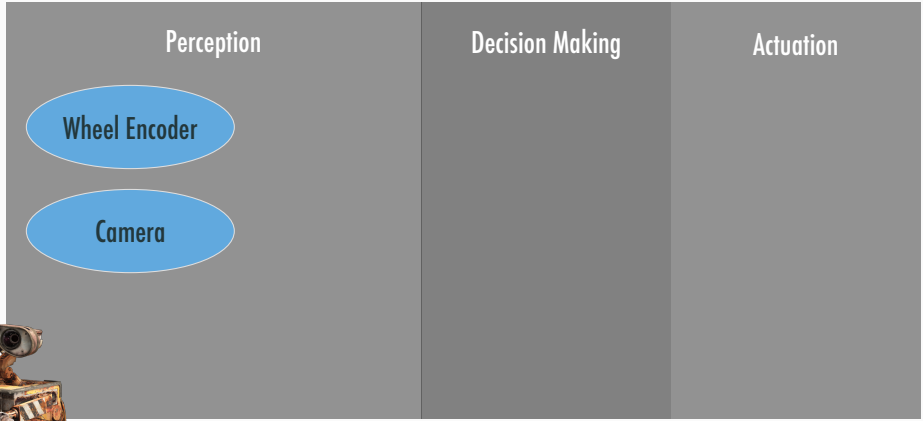
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



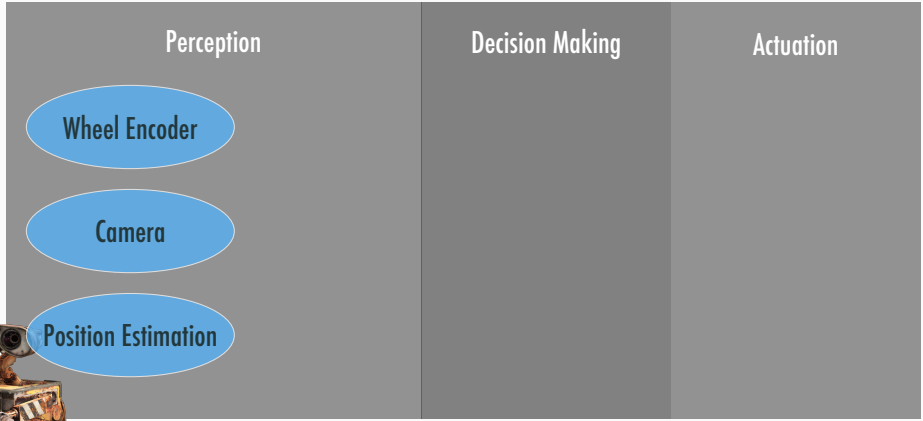
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



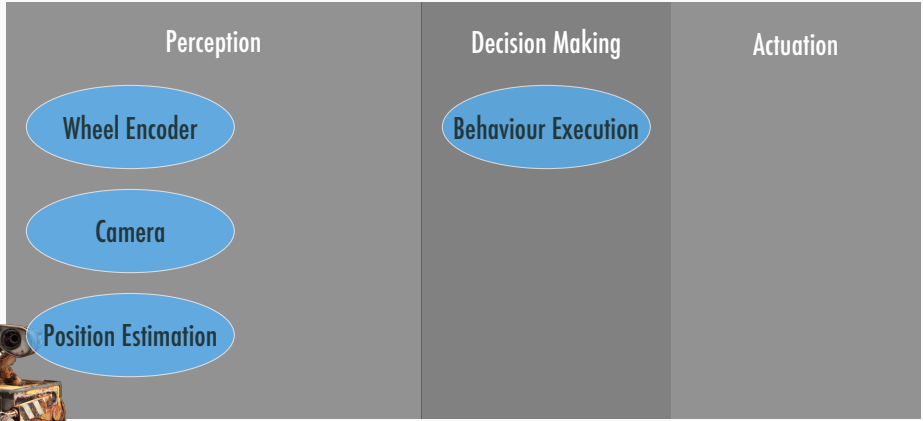
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



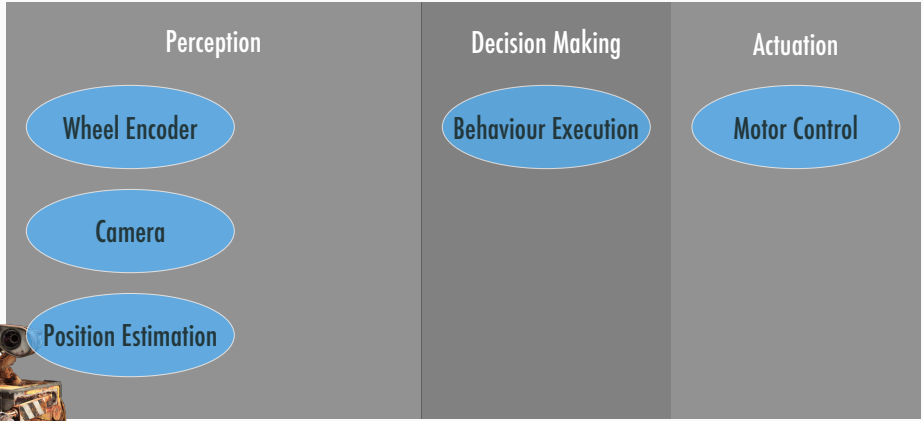
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



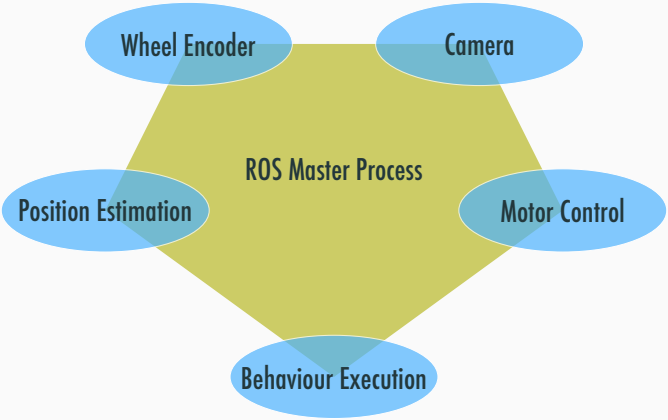
Nodes

ROS splits these high level tasks in low level ones and spawns a unix thread for each of them.



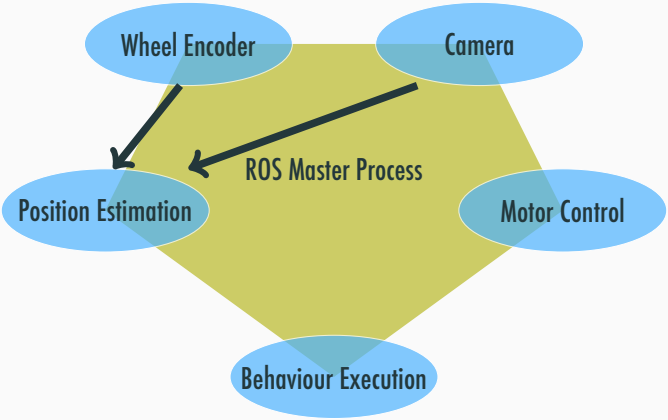
Nodes

The ROS Master Process is keeping a registry of all spawned nodes and allows them to establish communication between each other.



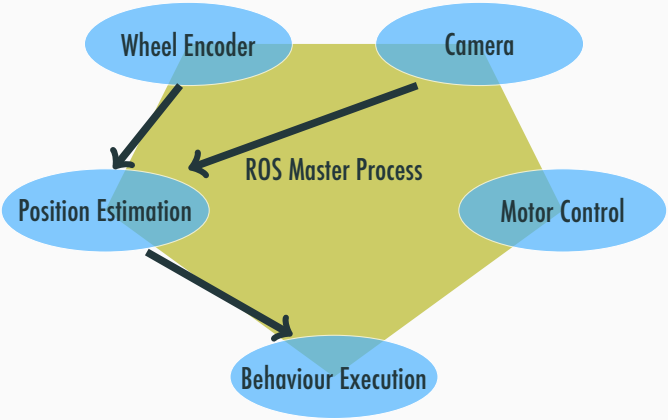
Nodes

The ROS Master Process is keeping a registry of all spawned nodes and allows them to establish communication between each other.



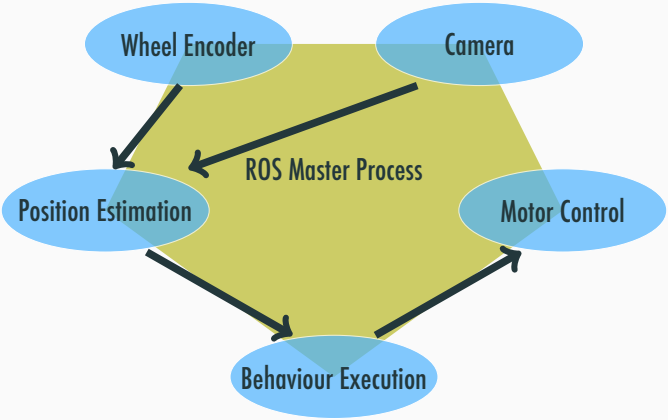
Nodes

The ROS Master Process is keeping a registry of all spawned nodes and allows them to establish communication between each other.



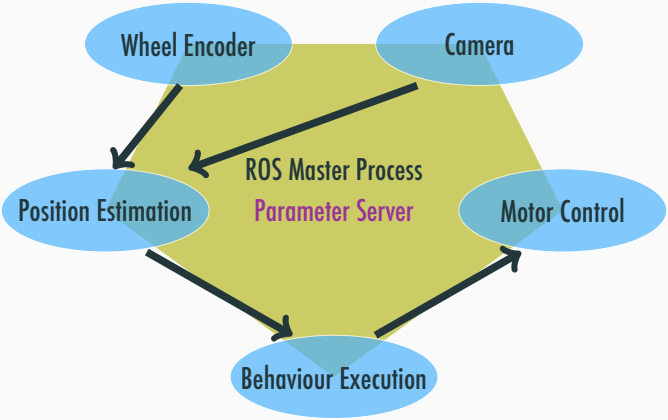
Nodes

The ROS Master Process is keeping a registry of all spawned nodes and allows them to establish communication between each other.



Nodes

The ROS Master Process is keeping a registry of all spawned nodes and allows them to establish communication between each other.



Topics

- ROS nodes communicate with each other over topics.

Topics

- ROS nodes communicate with each other over topics.
- If they want to send messages, they publish to a topic.

Topics

- ROS nodes communicate with each other over topics.
- If they want to send messages, they publish to a topic.
- If they want to receive messages, they subscribe to a topic.

Topics

- ROS nodes communicate with each other over topics.
- If they want to send messages, they publish to a topic.
- If they want to receive messages, they subscribe to a topic.

Messages

- ROS comes with a variety of predefined messages for:

Messages

- ROS comes with a variety of predefined messages for:
- Physical properties, e.g. positions, velocities, acceleration, rotations, durations

Messages

- ROS comes with a variety of predefined messages for:
- Physical properties, e.g. positions, velocities, acceleration, rotations, durations
- Sensor reading, e.g. laser scans, images, point clouds, inertial measurements

Messages

- ROS comes with a variety of predefined messages for:
- Physical properties, e.g. positions, velocities, acceleration, rotations, durations
- Sensor reading, e.g. laser scans, images, point clouds, inertial measurements
- There are over 200 different message types, but there is still the possibility to define custom ones

ROS Tutorials

Let’s get our hands dirty and our fingers on the keyboard.
ROS Tutorial

Racecar hacking

Some Notes

- The racecar should stop around 60cm in front of walls.

Some Notes

- The racecar should stop around 60cm in front of walls. (theoretically!)

Some Notes

- The racecar should stop around 60cm in front of walls. (theoretically!)
- Please open and close only the orange connectors.

Some Notes

- The racecar should stop around 60cm in front of walls. (theoretically!)
- Please open and close only the orange connectors.
- The speed is currently limited to 2m/s.

Some Notes

- The racecar should stop around 60cm in front of walls. (theoretically!)
- Please open and close only the orange connectors.
- The speed is currently limited to 2m/s. (currently ;))