AMR foundation Course

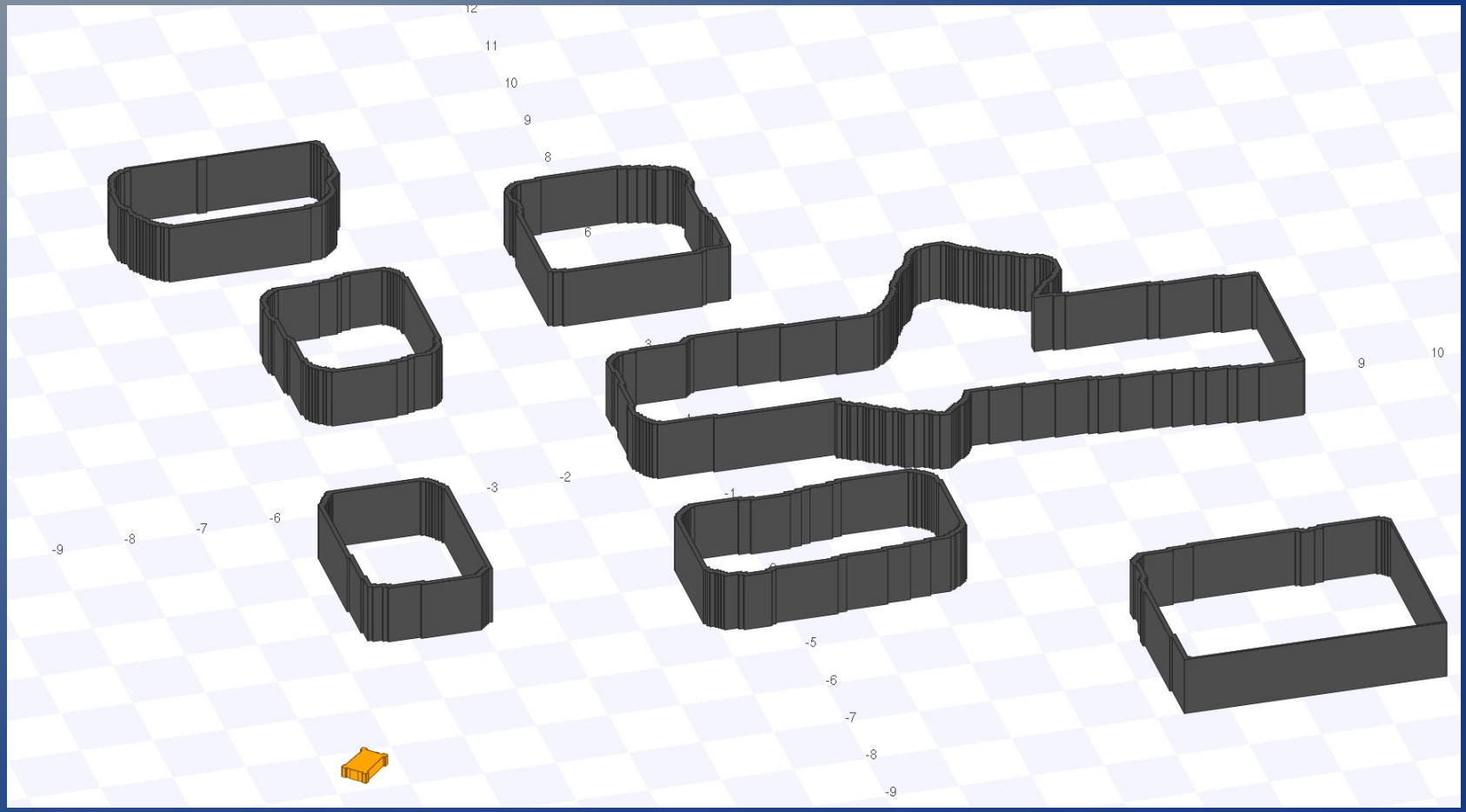
Robot Operating System(ROS)
Basic concepts and their usage
to prepare softwares
for your robots

Shehzad Ahmed
Master of Autonomous Systems
Hochschule Bonn-Rhein-Sieg
Sep. 12, 2014

ROS-Introduction

- First, lets see some cool videos and demos

AMR-Stage



ROS-Introduction

- It is like an operating system.
- Services of an operating system:
  - hardware abstraction
  - low-level device control,
  - implementation of commonly-used functionality,
  - message-passing between processes,
  - and process management.
- Similarly, ROS provides a 'communication infrastructure' to create a 'peer-peer network' of process.

ROS-Introduction

- Peer-Peer network of process?
- Let's understand this using a simple graph
  - Process/Processes? What is it?
  - Server? Why do we need this?
  - Data sharing/Message passing? But how?
    - Asynchronous
    - Synchronous
  - Process might also need configuration parameters.

ROS-Terminologies

- Process/Processes → Node/Nodes
- Server → ROS Master
- Message Passing:
  - Asynchronous → Topics
  - Synchronous → Services
- Config. Parameters → ROS parameter server

ROS-CONCEPTS

- ROS concepts are divided into three levels.
- ROS Computation Graph level(CGL)
  - Network of ROS processes that are performing data processing make a Computation graph.
  - Nodes,master, messages, topics services, parameters come under this level.

ROS CGL→ROS Master

- Two main responsibilities of ROS Master
  - Act as a nameserver therefore it provides name registration and lookup.
  - Acts as a parameter server.
- Nodes communicate with the master to:
  - Report their registration information.
  - Get registration information about other nodes.

ROS CGL → Nodes

- Nodes are processes that perform computation.
- Node-2-Node communication:
  - Server provides lookup information to all nodes.
  - Then Nodes directly create connection among them.
    - Using ROSTCP or ROSUDP protocols.
- Lookup information?
  - Names: They have a very important role in ROS.

ROS CGL → Messages

- Nodes communicate with each other by passing messages.

ROS CGL → Topics

- Topics acts as a message bus to transport them using publisher and subscriber model.
- Publisher and subscriber model uses only one type of message.
- Many-to-Many, one-way transport.

ROS CGL → Services

- Request and Response model is used by services.
- Services uses two types of messages: one for request and other for response.
- Its one-to-one two way communication.
- Client side operation: Request-Wait-Response pipeline.

ROS Concepts

- ROS Filesystem Level
  - This level more related to software organization of the ROS resources.
  - The concepts involved in this level are:
    - Packages: Organize ROS Software
    - MetaPackages: Combine Multiple Packages
    - Package Manifest: Provides package meta information and dependencies.
    - Repositories: Packages are stored and shared by using Version Control System(VCS).
- ROS Community Level

ROS-Practical Implementation

- ROS Computation Graph level concepts are implemented in ros_comm repository.
  - Provides communications-related packages
  - Stable supported core client libraries are roscpp and rospy.
  - Graph introspection tools (rostopic, rosnode, rosservice, rosparam).
  - Provide implementations and tools for CGL.

Navigating the ROS Filesystem

- Filesystem cmd Tools:
  - rospack
  - roscd
  - rosls

- Packages
  - Creating catkin packages
  - What makes up a catkin Package?
  - Catkin packages in a catkin Workspace.
  - Package dependencies
    - First-order dependencies
    - Indirect dependencies
  - Customizing the package.
  - Customizing the CmakeLists.txt.

ROS graph concepts

- ROS Nodes:
  - Nodes are written using ROS client libraries.
  - Creation of nodes CPP and Python.
- cmd tools:
  - Roscore: master+ rosout+parameter server
  - rosnode
  - Rosrun

- Initialization and Shutdown
  - Two levels of initialization.
    - Initializing the node through a call to one of the ros::init() functions.
    - Starting the node is most often done through creation of a ros::NodeHandle.
  - Node handle:
    - Automatic Startup and Shutdown of the node.
    - NodeHandles allows to specify a namespace.

- Callbacks and Spinning
  - Single-threaded Spinning:spin and spinOnce
  - Multi-threaded Spinning: MultiThreadedSpinner and ros::AsyncSpinner
- ROS Time and Duration
  - ROS has builtin time and duration primitive types.
  - ros::Time::now() and ros::Duration d(0.5)

- ROS Logging:
  - DEBUG,INFO,WARN,ERROR,FATAL

ROS graph concepts

- ROS Messages
  - ROS primitive types of messages
    - std_msgs
    - common_msgs
    - Custom type messages
  - Cmd tools
    - rosmsg ……
  - ROS message utilization.

ROS graph concepts

- ROS topics
  - nodes publish and subscribe desired topics.
  - topics transport messages.
  - topics publisher and subscriber node.
  - Cmd tools:
    - rostopic …..
    - rqt_graph
    - rqt_plot

- roslaunch tool:
  - Provides easiness to remap topic names.
  - Launch group of nodes to manage large network of nodes.
  - Roslaunch example.

ROS graph concepts

- ROS Services
  - Creating services.
  - Using services.
  - Service-Client pipeline.
  - CPP and python examples.

Tommorrow:Next to Come.....

- ROS parameter
- ros action server
- ROBOT description files ros transformation