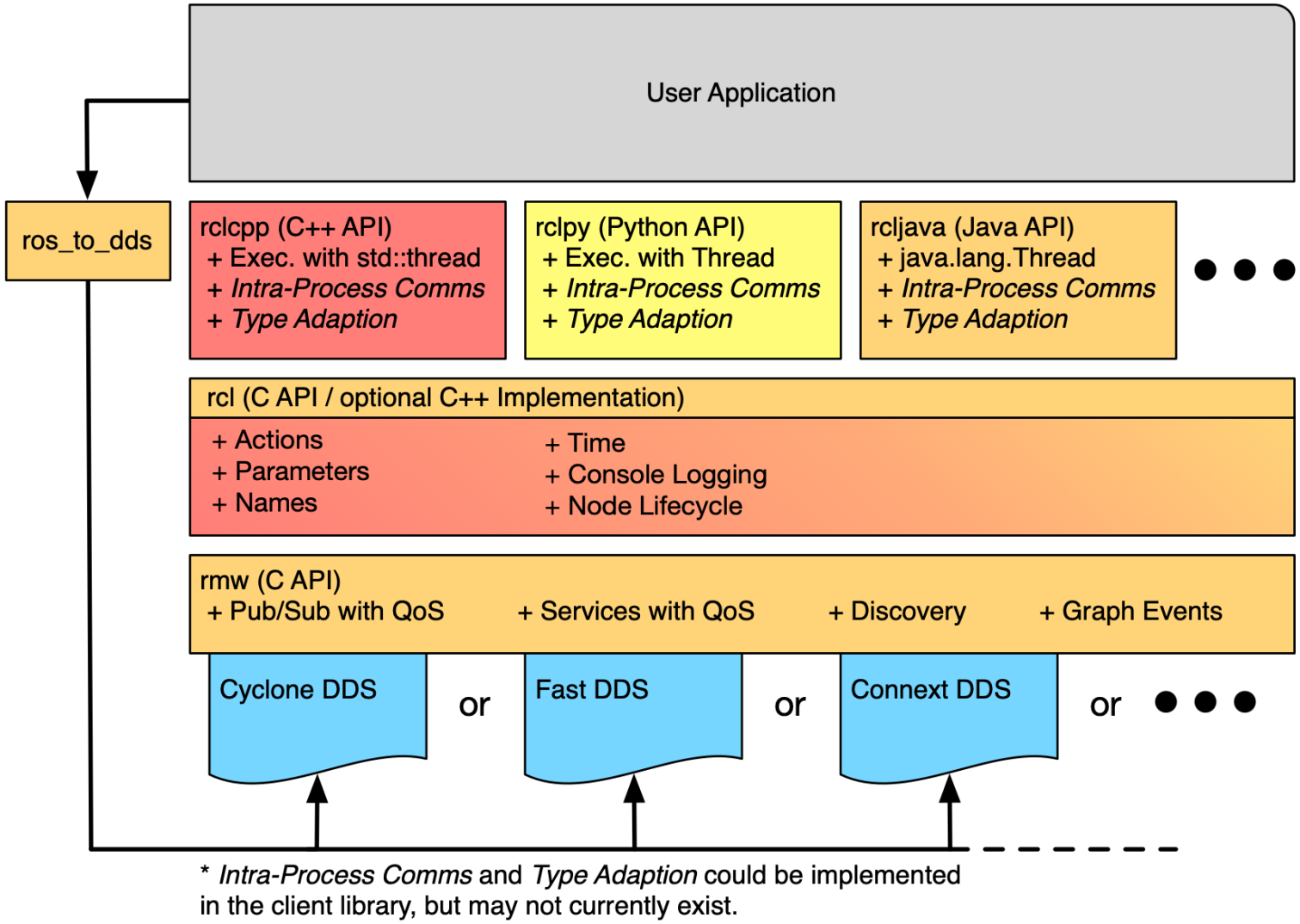
ROS2 – Part 2

Robotics and Computer Vision
BPC-PRP

Ing. Jakub Minařík
Brno University of Technology
2025

- DDS, Node Discovery, QoS
- Node Life Cycles, Parameters, Launchfiles, Executors
- ROS2 Packages
- RViz
- URDF
- TF2
- Simulators – Gazebo, Webots
- Bagfiles
- ROS2 Security

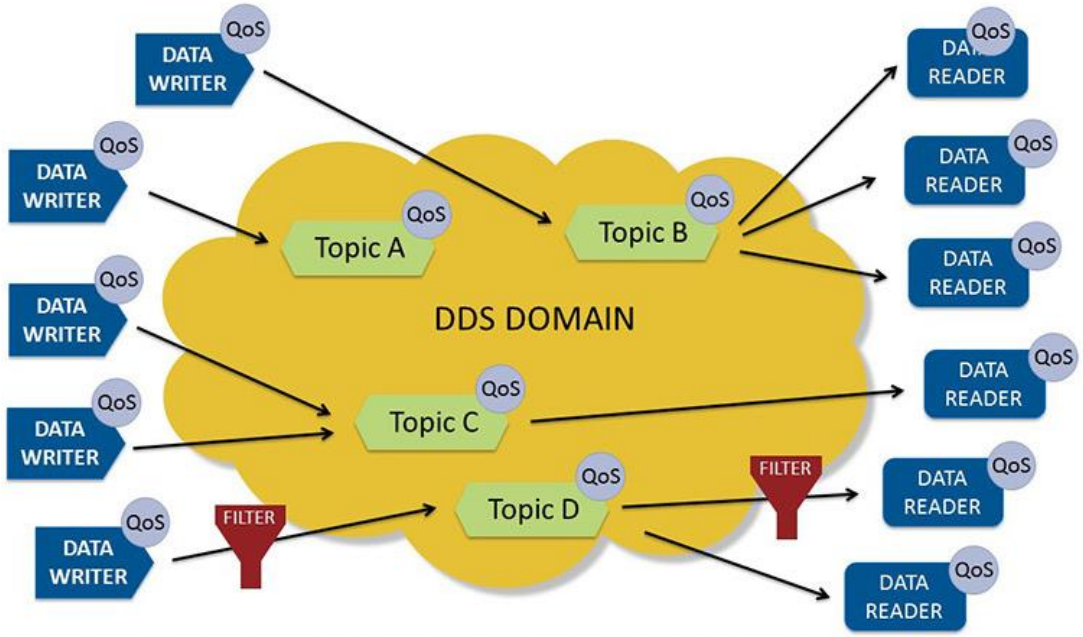
ROS2 API



1. https://docs.ros.org/en/humble/Concepts/Advanced/About-Internal-Interfaces.html

What is DDS?

- Middleware protocol and API standard
- Real-time data exchange, data-centric
- Publisher subscriber architecture
- Decentralized
- Quality of Service – QoS
- Multiple implementations - Fast DDS, Cyclone DDS, and RTI Connext...



1. https://www.dds-foundation.org/what-is-dds-3/

Node Discovery

- Handled by DDS implementation
- Basic idea:
  - When a node is started, it advertises its presence to other nodes on the network with the same ROS domain.
  - Nodes respond to this advertisement with information about themselves so that the appropriate connections can be made and the nodes can communicate.
  - Nodes periodically advertise their presence so that connections can be made with new-found entities, even after the initial discovery period.
  - Nodes advertise to other nodes when they go offline.
- Nodes make connection only if they have compatible QoS settings

1. https://docs.ros.org/en/humble/Concepts/Basic/About-Discovery.html

Quality of Service

- History
  - Keep last – store up to N samples
  - Keep all – depends on uderlying middleware
- Depth
  - Queue size – only when "Keep last" is set
- Reliability
  - Best effort – may lose samples
  - Reliable – may retry multiple times
- Durability
  - Transient local – publisher persist samples
  - Volatile – no attempt to persist samples
- Deadline
  - Duration – max time between messages
- Lifespan
  - Duration – max time message is considered valid
- Liveliness
  - Automatic – all publisher on one node are alive if one publish message
  - Manual by topic – only the publisher
- Lease Duration
  - Duration – max period of time publisher has to indicate its alive

1. https://docs.ros.org/en/humble/Concepts/Intermediate/About-Quality-of-Service-Settings.html

Launch File

- Starting multiple nodes
- Setting specific paremeters on start
- Reproducibility
- Substitutions
- Events
- Python, YAML, XML

Example Launch File

```
import launch
from launch_ros.actions import Node

def generate_launch_description():
  return launch.LaunchDescription([
    Node(
      package="turtlesim",
      executable="turtlesim_node",
      name="turtlesim1"
    )])
```

```
Node(
  package="turtlesim",
  executable="turtlesim_node",
  name="turtlesim2",
  remappings=[
    ("/turtle2/cmd_vel", "/turtle1/cmd_vel"),
    ("/turtle2/pose", "/turtle1/pose")
  ]
)
```

Parameters

- Associated with single node
- Configurable settings of node
- Can be changed on runtime
- Some are read-only
- Export/import as YAML
- In code change handled by callbacks

1. https://docs.ros.org/en/humble/Package-Docs.html

ros2 param --help

Various param related sub-commands

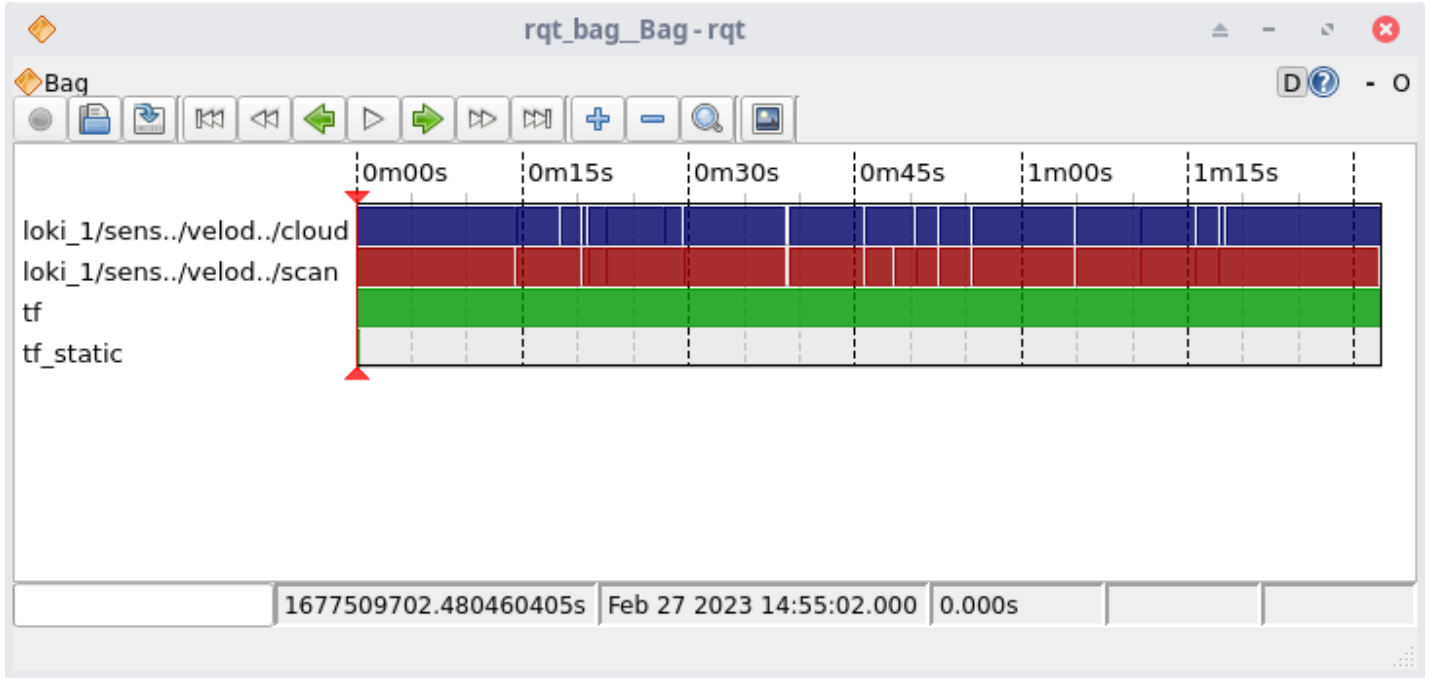
- options:
  - -h, --help show this help message and exit
- Commands:
  - delete Delete parameter
  - describe Show descriptive information about declared parameters
  - dump Show all of the parameters of a node in a YAML file format
  - get Get parameter
  - list Output a list of available parameters
  - load Load parameter file for a node
  - set Set parameter

Call `ros2 param <command> -h` for more detailed usage.

1. ROS2 humble command help - ros2 param --help

Bag files

- Recording and replaying data
- Testing, simulations, analyze data
- .db3 - SQLite3 format
- `ros2 bag record /topic_name`
- `ros2 bag play <bagfile_name>`



Executors

- Manage execution of callbacks from nodes
- Handle message passing, timers, services, and actions
- Manage multi-threading and parallel execution
- Prevent blocking issues in complex systems
- Ensure efficient processing of node callbacks
- Single-threaded and Multi-threaded

1. https://docs.ros.org/en/humble/Concepts/Intermediate/About-Executors.html

Executors - issues

- Complex and mixed scheduling semantics.
- Higher priority callbacks may be blocked by lower priority callbacks.
- No explicit control over the callbacks execution order.
- No built-in control over triggering for specific topics.

Executor in C++


```
rclcpp::Node::SharedPtr node1 = ...
rclcpp::Node::SharedPtr node2 = ...

rclcpp::executors::SingleThreadedExecutor executor;

executor.add_node(node1);
executor.add_node(node2);

executor.spin();
```

1. https://docs.ros.org/en/humble/Concepts/Intermediate/About-Executors.html

Workspace

- Top directory
- Source directory containing ROS 2 packages
- Build directory (generated after colcon build)
- Install directory containing installed packages
- Logging directory (stores logs from executions)
- Metadata for colcon build tool (optional)

```
ros2_ws/
├── src/
├── build/
├── install/
├── log/
└── colcon.meta
```

Packages

- Nodes, launch files, configuration files, message definition and other resources
- Modular, maintainability, colaboration
- Officialy support for c++ and python
- Create package for c++
  - `cd ~/ros2_ws/src`
  - `ros2 pkg create my_package --build-type ament_cmake --dependencies rclcpp std_msgs`

Package structure

--build-type ament_cmake

```
my_package/
  ├─ CMakeLists.txt
  ├─ include/my_package/
  ├─ package.xml
  └─ src/
```

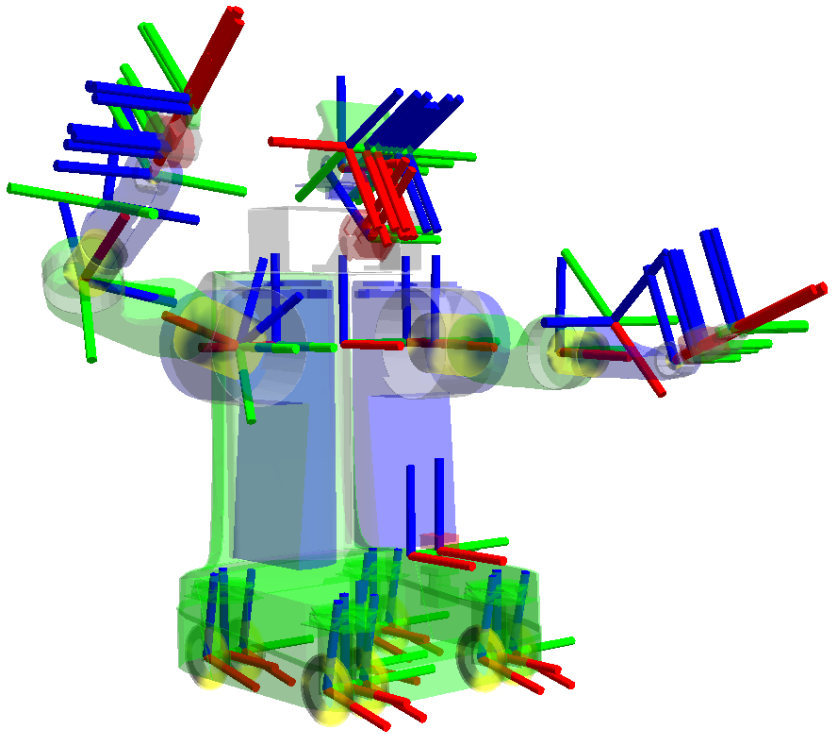
--build-type ament_python

```
my_package/
  ├─ package.xml
  ├─ resource/my_package
  ├─ setup.cfg
  ├─ setup.py
  └─ my_package/
```

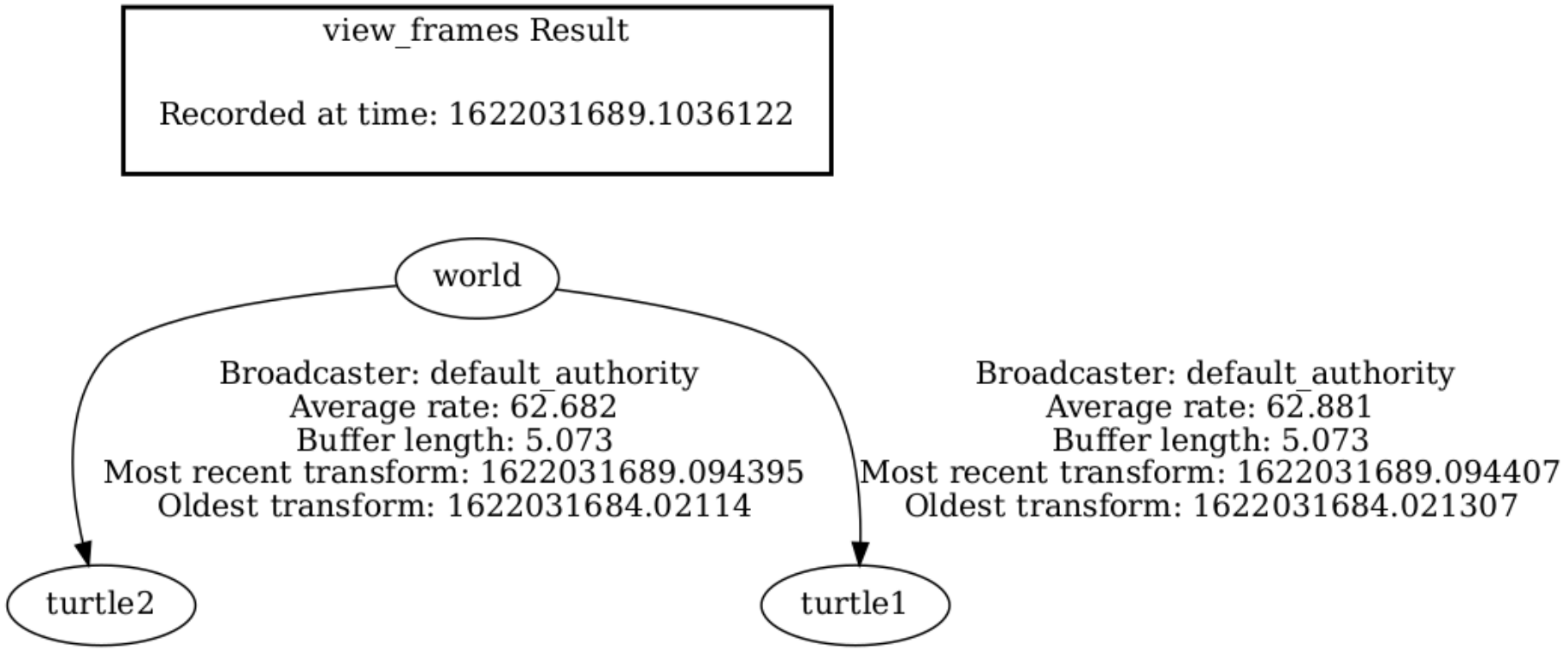
Packages dependencies

- package.xml
- CMakeLists.txt
- `rosdep`
  - Manage dependencies
  - Abstract uderlying packages managers
  - Install dependencies from `package.xml`
  - `rosdep install --from-paths src --ignore-src -r -y`

- Multiple coordinates frames
- Standart way to transform data between frames
- Transform define relation between frame
- Each transform has parent and child frame – creating tree
- Static dynamic transforms



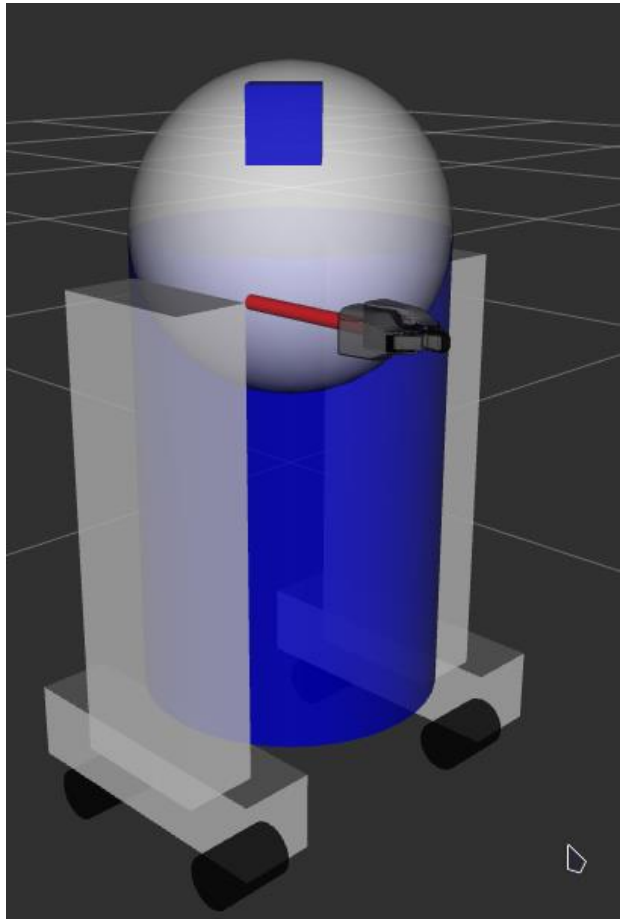
1. https://docs.ros.org/en/humble/Concepts/Intermediate/About-Tf2.html



1. https://docs.ros.org/en/humble/Tutorials/Intermediate/Tf2/Introduction-To-Tf2.html

URDF

- Unified Robot Description Format
- XML
- Represents robot model - defines
  - Links
  - Joints
  - Physical properties
- Imported to simulators and RViz2
- Many conversions tools from 3D models

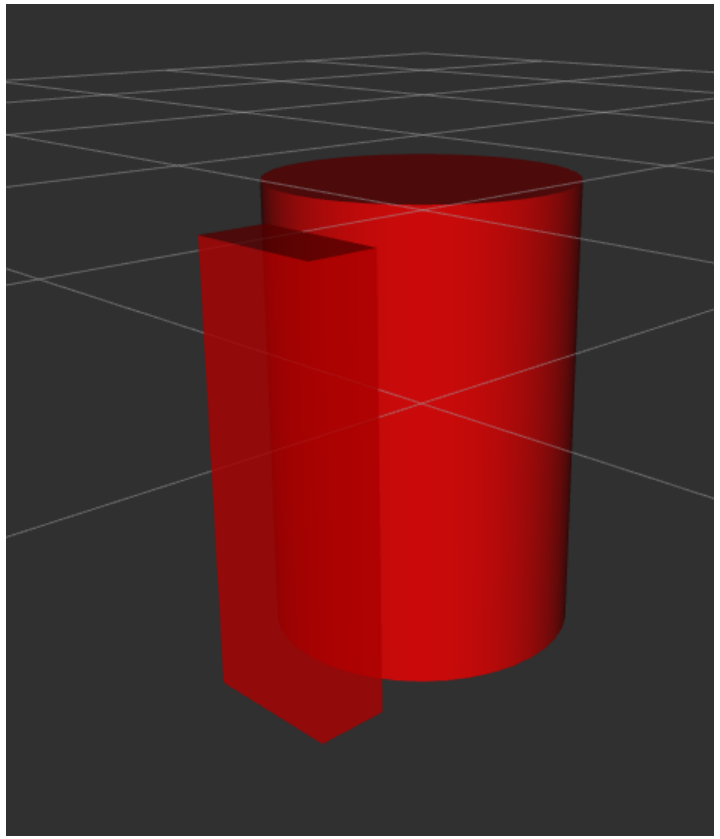


1. https://docs.ros.org/en/humble/Tutorials/Intermediate/URDF/Building-a-Movable-Robot-Model-with-URDF.html

URDF

```
<?xml version="1.0"?>
<robot name="origins">
  <link name="base_link">
    <visual>
      <geometry>
        <cylinder length="0.6" radius="0.2"/>
      </geometry>
    </visual>
  </link>

  <link name="right_leg">
    <visual>
      <geometry>
        <box size="0.6 0.1 0.2"/>
      </geometry>
      <origin rpy="0 1.57075 0" xyz="0 0 -0.3"/>
    </visual>
  </link>
```



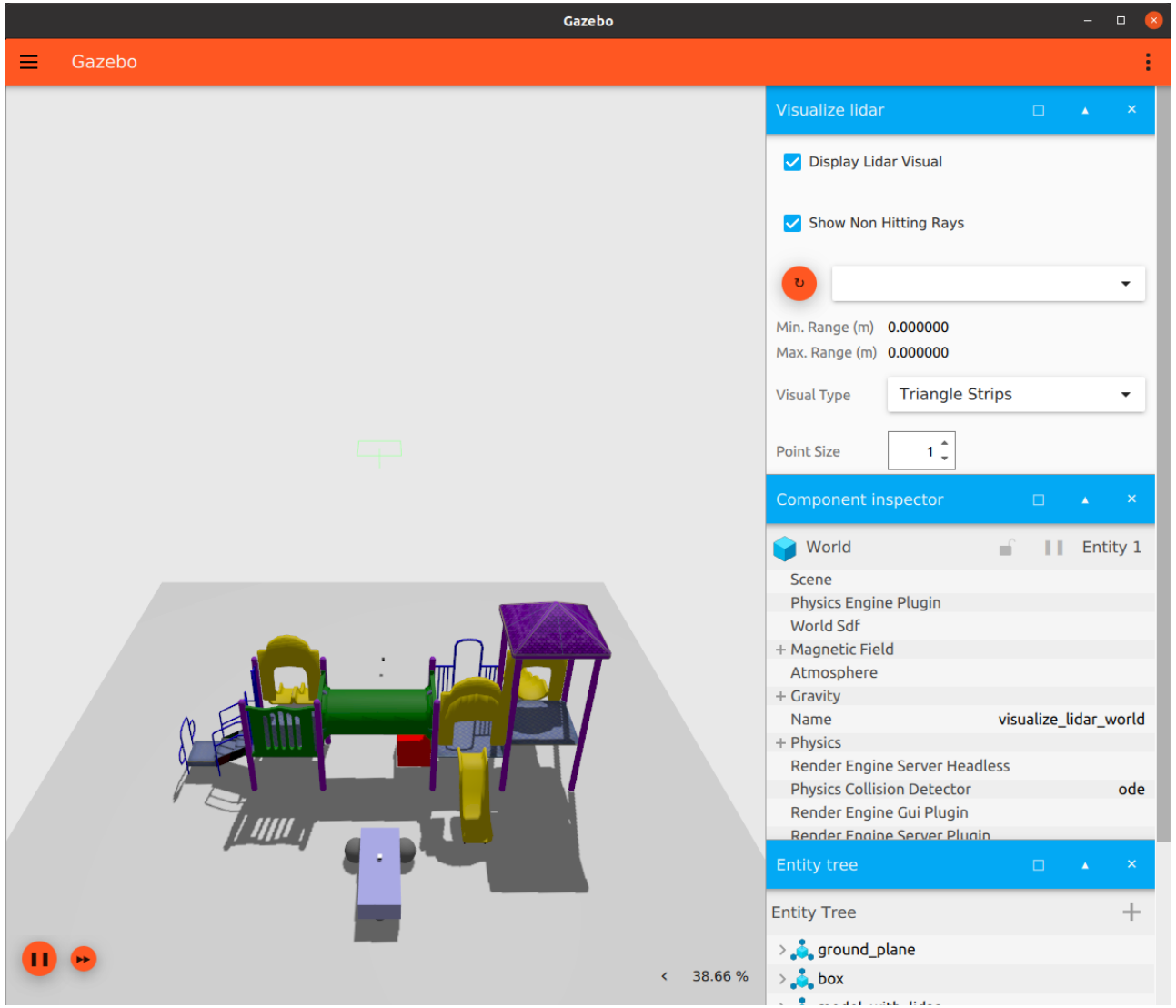
1. https://docs.ros.org/en/humble/Tutorials/Intermediate/URDF/Building-a-Visual-Robot-Model-with-URDF-from-Scratch.html

- 3D visualizer
- Out-of-box support for usually used data types – images, point clouds, transformations,…
- Configuration files – can be loaded on start from launch file
- Markers
  - Custom visualization objects/elements
  - Many different shapes or can use render
- URDF

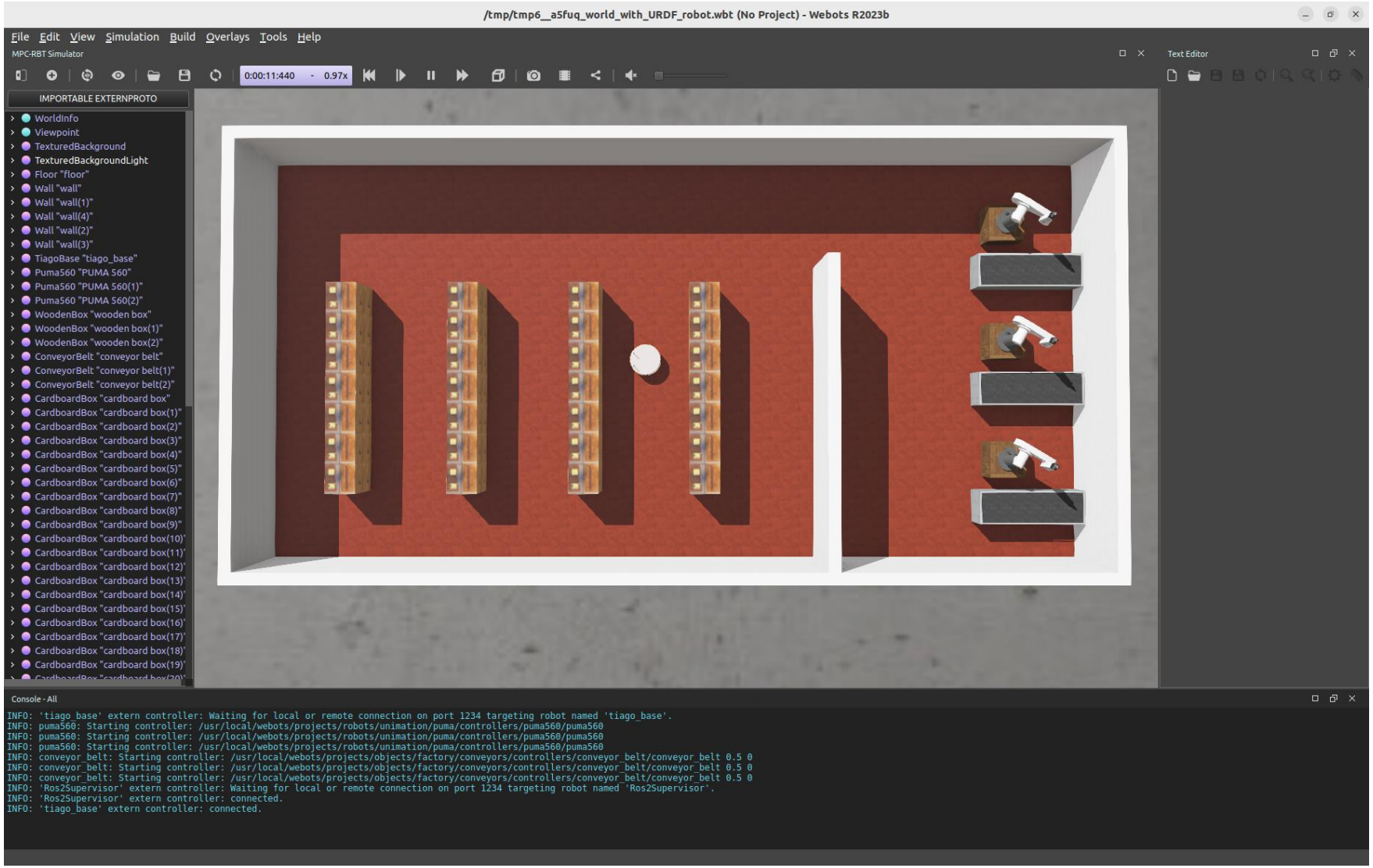
Simulators

- Gazebo Ignition, Webots
- Support URDF
- Testing without hardware
- Sensors support
- Physical integration

1. https://docs.ros.org/en/humble/Tutorials/Advanced/Simulators/Simulation-Main.html



1. https://docs.ros.org/en/humble/Tutorials/Advanced/Simulators/Gazebo/Gazebo.html


/tmp/tmp6__a5fuq_world_with_URDF_robot.wbt (No Project) - Webots R2023b
File Edit View Simulation Build Overlays Tools Help
MPC-RBT Simulator
0:00:11:440 - 0.97x
Text Editor
IMPORTABLE EXTERNPROTO
WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
Floor "floor"
Wall "wall"
Wall "wall(1)"
Wall "wall(4)"
Wall "wall(2)"
Wall "wall(3)"
TiagoBase "tiago_base"
Puma560 "PUMA 560"
Puma560 "PUMA 560(1)"
Puma560 "PUMA 560(2)"
WoodenBox "wooden box"
WoodenBox "wooden box(1)"
WoodenBox "wooden box(2)"
ConveyorBelt "conveyor belt"
ConveyorBelt "conveyor belt(1)"
ConveyorBelt "conveyor belt(2)"
CardboardBox "cardboard box"
CardboardBox "cardboard box(1)"
CardboardBox "cardboard box(2)"
CardboardBox "cardboard box(3)"
CardboardBox "cardboard box(4)"
CardboardBox "cardboard box(5)"
CardboardBox "cardboard box(6)"
CardboardBox "cardboard box(7)"
CardboardBox "cardboard box(8)"
CardboardBox "cardboard box(9)"
CardboardBox "cardboard box(10)"
CardboardBox "cardboard box(11)"
CardboardBox "cardboard box(12)"
CardboardBox "cardboard box(13)"
CardboardBox "cardboard box(14)"
CardboardBox "cardboard box(15)"
CardboardBox "cardboard box(16)"
CardboardBox "cardboard box(17)"
CardboardBox "cardboard box(18)"
CardboardBox "cardboard box(19)"
Console - All
INFO: 'tiago_base' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'tiago_base'.
INFO: puma560: Starting controller: /usr/local/webots/projects/robots/unimation/puma/controllers/puma560/puma560
INFO: puma560: Starting controller: /usr/local/webots/projects/robots/unimation/puma/controllers/puma560/puma560
INFO: puma560: Starting controller: /usr/local/webots/projects/robots/unimation/puma/controllers/puma560/puma560
INFO: conveyor_belt: Starting controller: /usr/local/webots/projects/objects/factory/conveyors/controllers/conveyor_belt/conveyor_belt 0.5 0
INFO: conveyor_belt: Starting controller: /usr/local/webots/projects/objects/factory/conveyors/controllers/conveyor_belt/conveyor_belt 0.5 0
INFO: conveyor_belt: Starting controller: /usr/local/webots/projects/objects/factory/conveyors/controllers/conveyor_belt/conveyor_belt 0.5 0
INFO: 'Ros2Supervisor' extern controller: Waiting for local or remote connection on port 1234 targeting robot named 'Ros2Supervisor'.
INFO: 'Ros2Supervisor' extern controller: connected.
INFO: 'tiago_base' extern controller: connected.

Jakub Minařík
203294@vut.cz

Brno University of Technology
Faculty of Electrical Engineering and Communication
Department of Control and Instrumentation

Robotics and AI Research Group