

# Моделирование колесных роботов

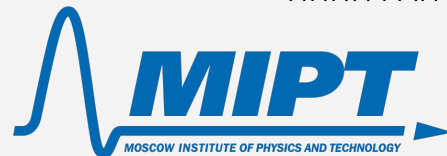
Лекция 11. Имена и пространства имен в  
ROS, время, инструменты отладки и  
симуляции

Николай Жердев

Москва, 2023



ИППИ РАН





# СОДЕРЖАНИЕ ЛЕКЦИИ

1. Имена и пространства имен в ROS
2. Время
3. Дополнительные инструменты
  - a. tf
  - b. .bag файлы
  - c. Rqt
  - d. Rviz
  - e. Gazebo

# ИМЕНА В ROS

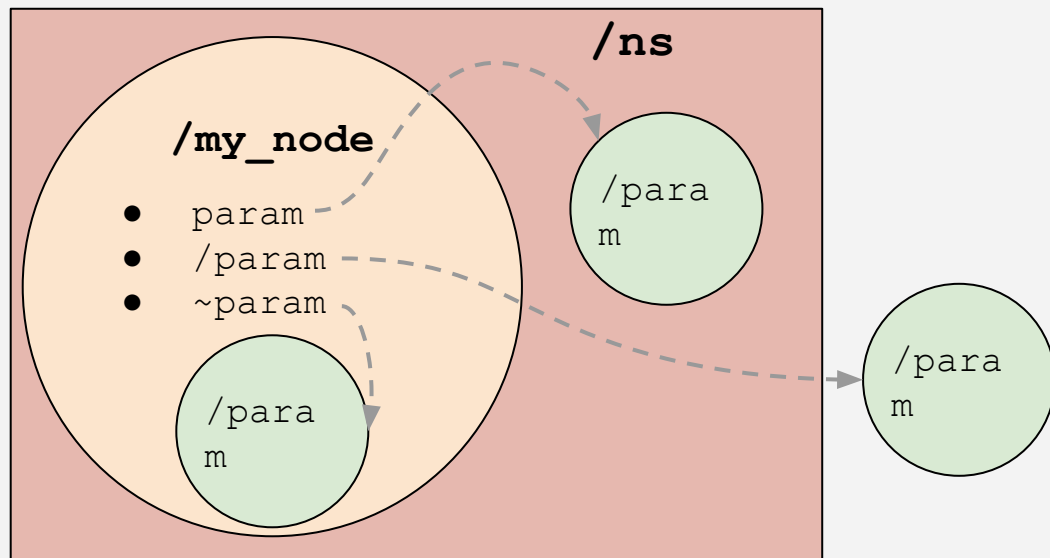
<http://wiki.ros.org/Names>

- ❑ В ROS имена есть у:
  - ❑ Нодов
  - ❑ Топиков
  - ❑ Параметров
  - ❑ Сервисов
  - ❑ ...
- ❑ Имена и пространства имен позволяют инкапсулировать информацию
- ❑ Валидные имена отвечают следующим правилам:
  - ❑ Первый символ: ([a-zA-Z]), тильда (~) или прямой слеш (/)
  - ❑ Последующие символ: ([0-9a-zA-Z]), подчеркивания (\_), или прямые слешы (/)

# ИМЕНА В ROS

<http://wiki.ros.org/Names>

- ❑ Имя может иметь один из 4 типов:
  - ❑ relative/name
    - ❑ base
  - ❑ /global/name
  - ❑ ~private/name
- ❑ Правила преобразования имен на примере нода и параметра:



Имя нода	Относительное имя	Глобальное имя	Приватное имя
/my_node	param -> /param	/param -> /param	~param -> /my_node/param
/ns/my_node	param -> /ns/param	/param -> /param	~param -> /ns/my_node/param
	param_ns/param -> /ns/param_ns/param	/param_ns/param -> /param_ns/param	~param_ns/param -> /ns/my_node/param_ns/param

# ВРЕМЯ В ROS

<http://wiki.ros.org/rospy/Overview/Time>

- ❑ UNIX-время используется в ROS в качестве временных меток
  - ❑ **UNIX-время** — целое число, увеличивающееся каждую секунду и равное количеству секунд, прошедших с 00:00:00 UTC 1 января 1970 года
- ❑ Клиентские библиотеки (rospy, roscpp, ...) предоставляют API для работы со временем:
  - ❑ Базовые классы `Time` и `Duration`, `Timer` с поддержкой арифметических операций
  - ❑ Функции для получения системного времени
  - ❑ Функции `rospy.sleep()` и `rospy.Rate.sleep()`

```
now = rospy.get_rostime() # эквивалентно now = rospy.Time.now()
rospy.loginfo("Current time %i %i", now.secs, now.nsecs)
```

```
two_hours = rospy.Duration(60*60) + rospy.Duration(60*60)
one_hour = rospy.Duration(0*60*60) - rospy.Duration(60*60)
tomorrow = rospy.Time.now() + rospy.Duration(24*60*60)
negative_one_day = rospy.Time.now() - tomorrow
```

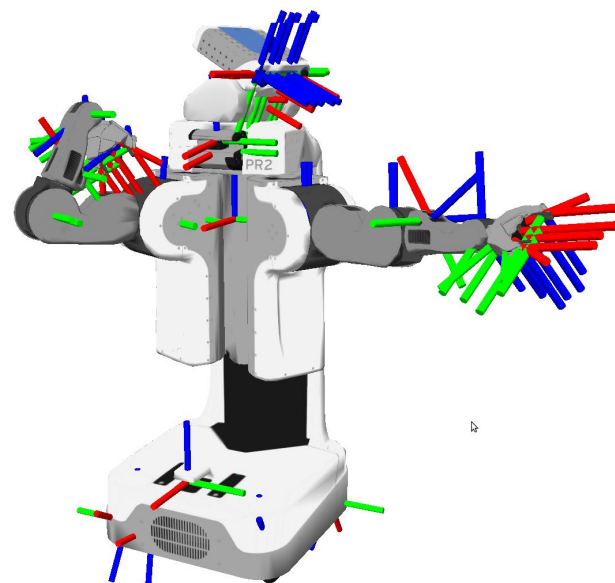
```
# sleep for 10 seconds
rospy.sleep(10.)
# sleep for duration
d = rospy.Duration(10, 0)
rospy.sleep(d)
```

# Tf

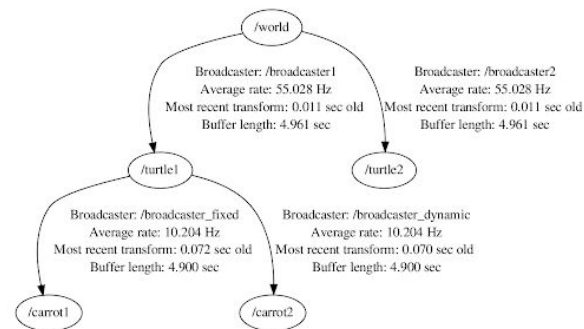
<http://wiki.ros.org/tf>

<http://wiki.ros.org/tf2>

- ❑ Пакеты tf и tf2:
  - ❑ Хранят преобразования между системами координат робота в виде дерева буферизованного во времени
  - ❑ Предоставляют интерфейс для преобразования точек векторов и т.д. между любыми системами координат
  - ❑ Позволяют восстановить преобразования между системами координат в прошлом



view\_frames Result  
Recorded at time: 1254266776.683



# ROSBAG

<http://wiki.ros.org/rosbag>

Что у тебя в сумке? Как работать с bag файлами и не утонуть в данных

- ❑ .bag файлы содержат **сериализованные** ROS-сообщения.
- ❑ .bag файлы могут быть проиграны в виде топиков, которые в них записаны.
- ❑ Формат .bag файлов **эффективен** как **для записи**, так и **для проигрывания**, так как сообщения хранятся в том же формате, что и при передаче по сети внутри ROS.

# КАК ЗАПИСАТЬ .BAG ФАЙЛ

`rosvag record <topic-names>`

`-d, --duration`

Максимальная продолжительность файла.

```
$ rosvag record --duration=30 /chatter
```

```
$ rosvag record --duration=5m /chatter
```

```
$ rosvag record --duration=2h /chatter
```

`--split`

Разделение файла когда достигнута максимальная длина/продолжительность

```
$ rosvag record --split --size=1024 /chatter
```

```
$ rosvag record --split --duration=30 /chatter
```

```
$ rosvag record --split --duration=5m /chatter
```

```
$ rosvag record --split --duration=2h /chatter
```

`--max-splits=MAX_SPLITS`

Разделить bag максимум **MAX\_SPLITS** раз, после чего начать удалять устаревшие файлы.

```
$ rosvag record --split --size 1024 --max-splits 3 /chatter
```

```
$ rosvag record --split --duration 10m --max-splits 6 /chatter
```

`-b SIZE, --buffsize=SIZE`

Использовать внутренний буфер размера SIZE MB (Default: 256, 0 = бесконечный). Создает очередь сообщений объекта recorder, которая заполняется до того как быть записанов в файл.

Уменьшения размера буфера приведет к потере сообщений.

```
$ rosvag record -b 1024 /chatter
```

`--chunksize=SIZE`

Записывать блоки данных размера SIZE KB (Default: 768). Это размер буфера объекта bag файл. Уменьшение буфера приведет к более частой записи на диск.

```
$ rosvag record --chunksize=1024 /chatter
```

`-l NUM, --limit=NUM`

Записать только NUM сообщений из каждого топика.

```
$ rosvag record -l 1000 /chatter
```

`--node=NODE`

Записать все топики, на которые подписан NODE.

```
$ rosvag record --node=/joy_teleop
```

`-j, --bz2`

Применить компрессию BZ2.

```
$ rosvag record -j /chatter
```

`--lz4`

Применить компрессию LZ4.

```
$ rosvag record --lz4 /chatter
```



# КАК ВОСПРОИЗВЕСТИ .BAG ФАЙЛ

`rosbag play <bag-files>` - считывает содержимое bag файлов и публикует их синхронизируя по времени

`-i, --immediate`

Проиграть все сообщения без задержки.

`$ rosbag play -i recorded1.bag`

`--pause`

Начать проигрывание в режиме "Пауза".

`$ rosbag play --pause recorded1.bag`

`--queue=SIZE`

Размер очереди публикации сообщений SIZE (по умолчанию: 0).

`$ rosbag play --queue=1000 recorded1.bag`

`--clock`

Публиковать время.

`$ rosbag play --clock recorded1.bag`

`--hz=HZ`

Публиковать время с частотой HZ (по умолчанию: 100).

`$ rosbag play --clock --hz=200 recorded1.bag`

`-d SEC, --delay=SEC`

Делать паузу в проигрывании на SEC секунд после каждого нового объявления топика (чтобы подписчики имели время на подписку).

`$ rosbag play -d 5 recorded1.bag`

`-r FACTOR, --rate=FACTOR`

Умножить скорость воспроизведения на FACTOR.

`$ rosbag play -r 10 recorded1.bag`

`-s SEC, --start=SEC`

Начать воспроизведение с SEC секунды от начала файла.

`$ rosbag play -s 5 recorded1.bag`

`-u SEC, --duration=SEC`

Проиграть SEC секунд файла.

`$ rosbag play -u 240 recorded1.bag`

`--skip-empty=SEC`

Пропустить участки bag файла без сообщений длиной более SEC секунд.

`$ rosbag play --skip-empty=1 recorded1.bag`

`-l, --loop`

Зациклить воспроизведение.

`$ rosbag play -l recorded1.bag`

`-k, --keep-alive`

Не останавливать воспроизведение после проигрывания всех сообщений (полезно для публикации latched топиков).

`$ rosbag play -k recorded1.bag`

# ПРОСМОТР СОДЕРЖИМОГО

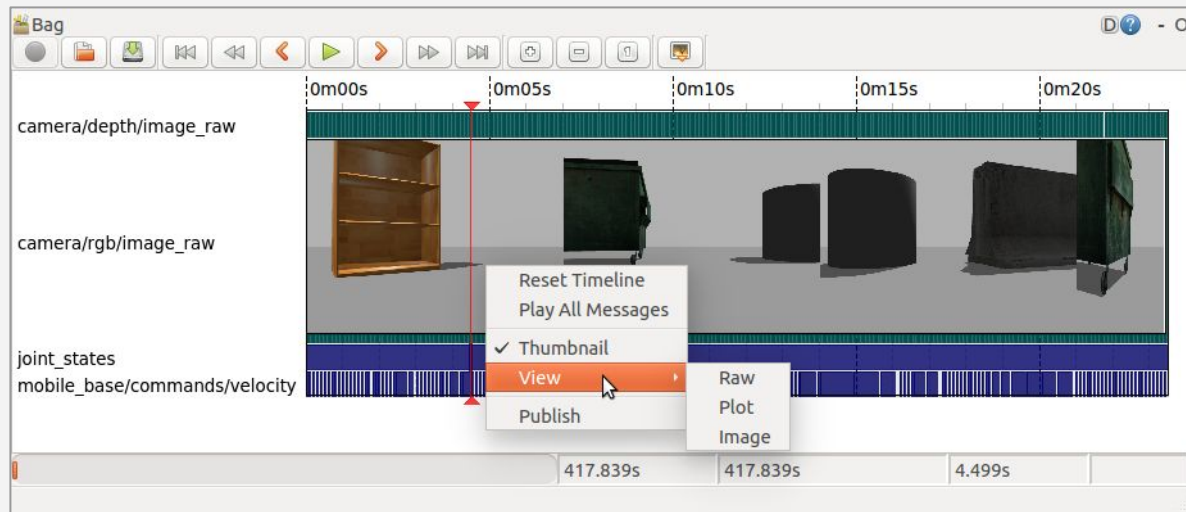
- ❑ `roscat info <bag-files>`
- ❑ `rostopic list -b <bag-file>`
- ❑ `rostopic echo -b <bag-file>`

```
shipltko@devel-Latitude-5491: ~
+ ~ rosbag info 2018-11-07-16-37-15_0.bag
path: 2018-11-07-16-37-15_0.bag
version: 2.0
duration: 2:06s (126s)
start: Nov 07 2018 16:37:15.38 (1541597835.38)
end: Nov 07 2018 16:39:21.63 (1541597961.63)
size: 272.6 MB
messages: 68604
compression: none [325/325 chunks]
types:
  al6_vision_msgs/RoadRecognitionResult [b04d56e623d6211527e5deac7cbe334c]
  geometry_msgs/PoseStamped [d3812c3cbc69362b77dc0b19b345f8f5]
  motion_control_msgs/ControlMode [49e73f9e9ca1259cca296097ee6ca9fd]
  motion_control_msgs/TopLevelControllerState2 [2753d177e7e10a80f0bfc696fc462d63]
  motion_control_msgs/VehicleDriverState [fdeb54fd80579e12f224d3939cb2b51]
  motion_control_msgs/WaypointArray2 [a254e1483dd343b241d358c8e42b60b]
  rosbag_msgs/Log [acfffd30cd6b6de30f120938c17c593fb]
  sensor_msgs/CameraInfo [c9a58c1b0b154e0e6da757cb991d1214]
  sensor_msgs/CompressedImage [8f7a12909da2c9d3332d540a977563f]
  sensor_msgs/Imu [6a62c6daae103f4ff57a132d6f95cec2]
  sensor_msgs/TimeReference [fdeb64a0265108ba86c3d38fb11c0c16]
  tf2_msgs/TFMessage [94810edda583a504dfda3829e70d7ec]
  vi_device_msgs/DriverTask [2d67f2c905e96514589d2d04684e134]
  vi_device_msgs/OdometryExtended [7b17d630a932b60214780f1b41298095]
  vi_device_msgs/RotationSensors2 [ffdd143616a45ccb7e887fd9c143fc8cb]
  vi_device_msgs/SyncMultiRange [7f6d9e2826036f9f336e115e5206396c]
  vi_device_msgs/VehicleTask2v3 [7d89c1149e46ee01b1bfec956a897d13]
  vi_nmea_msgs/Sentence [9f221efc5f4b3ac7ce4af10b32308b]
  vi_nmea_navsat_driver/NavSatFixExtended [850aad466c4e594402c70781e9317ef]
  walls_detection/WallsRecognitionResult [b1745e055246a775d31769d0117453aa]
types:
  /control/control_mode 1261 msgs : motion_control_msgs/ControlMode
  /control/top_level_control/state 1261 msgs : motion_control_msgs/TopLevelControllerState2
  /control/trajectory/waypoints 1261 msgs : motion_control_msgs/WaypointArray2
  /depth/depth_registered/compressedDepth 336 msgs : sensor_msgs/CompressedImage
  /driver/task 7355 msgs : vi_device_msgs/DriverTask
  /gnss/fix 126 msgs : vi_nmea_navsat_driver/NavSatFixExtended
  /gnss/nmea_sentence 792 msgs : vi_nmea_msgs/Sentence
  /gnss/time_reference 126 msgs : sensor_msgs/TimeReference
  /imu/xsens/imu 5045 msgs : sensor_msgs/Imu
  /left/camera_info 337 msgs : sensor_msgs/CameraInfo
  /left/image_rect_color/compressed 343 msgs : sensor_msgs/CompressedImage
  /map/pose 1261 msgs : geometry_msgs/PoseStamped
  /odometry/extended 12870 msgs : vi_device_msgs/OdometryExtended
  /odometry/rear_wheels2 7824 msgs : vi_device_msgs/RotationSensors2
  /pc/detector/walls 252 msgs : walls_detection/WallsRecognitionResult
  /right/camera_info 336 msgs : sensor_msgs/CameraInfo
  /right/image_rect_color/compressed 339 msgs : sensor_msgs/CompressedImage
  /rosout 59 msgs : rosbag_msgs/Log
(17 connections)
  /rosout_agg 29 msgs : rosbag_msgs/Log
  /sonars/front/sync 1232 msgs : vi_device_msgs/SyncMultiRange
  /tf 18254 msgs : tf2_msgs/TFMessage
(4 connections)
  /vehicle/state 2523 msgs : motion_control_msgs/VehicleDriverState
  /vehicle/task 2523 msgs : vi_device_msgs/VehicleTask2v3
  /vision/front/right/camera_info 1261 msgs : sensor_msgs/CameraInfo
  /vision/front/right/image/compressed 1261 msgs : sensor_msgs/CompressedImage
  /vision/front/right/road_recognition 337 msgs : al6_vision_msgs/RoadRecognitionResult
```

# ПРОСМОТР СОДЕРЖИМОГО

## Пакет `rqt_bag`

- ❑ Показывает наличие сообщений в топиках
- ❑ Показывает thumbnails изображения на временной шкале
- ❑ Позволяет строить графики числовых сообщений
- ❑ Публиковать / записывать выбранные топики
- ❑ Экспортировать сообщения из выбранного временного промежутка в новый bag



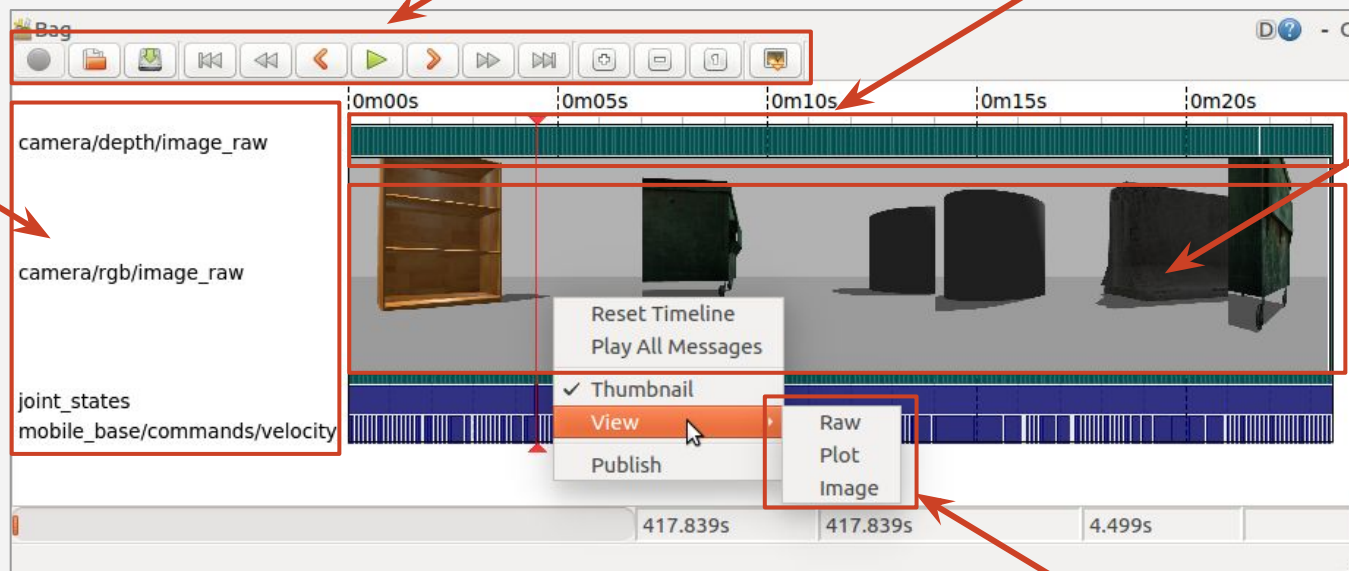
`rqt_bag` имеет API, позволяющий реализовывать свои плагины

Воспроизведение/запись/навигация

Индикация прихода сообщений в топик

Список  
топиков

Визуализация  
топиков с  
изображениями



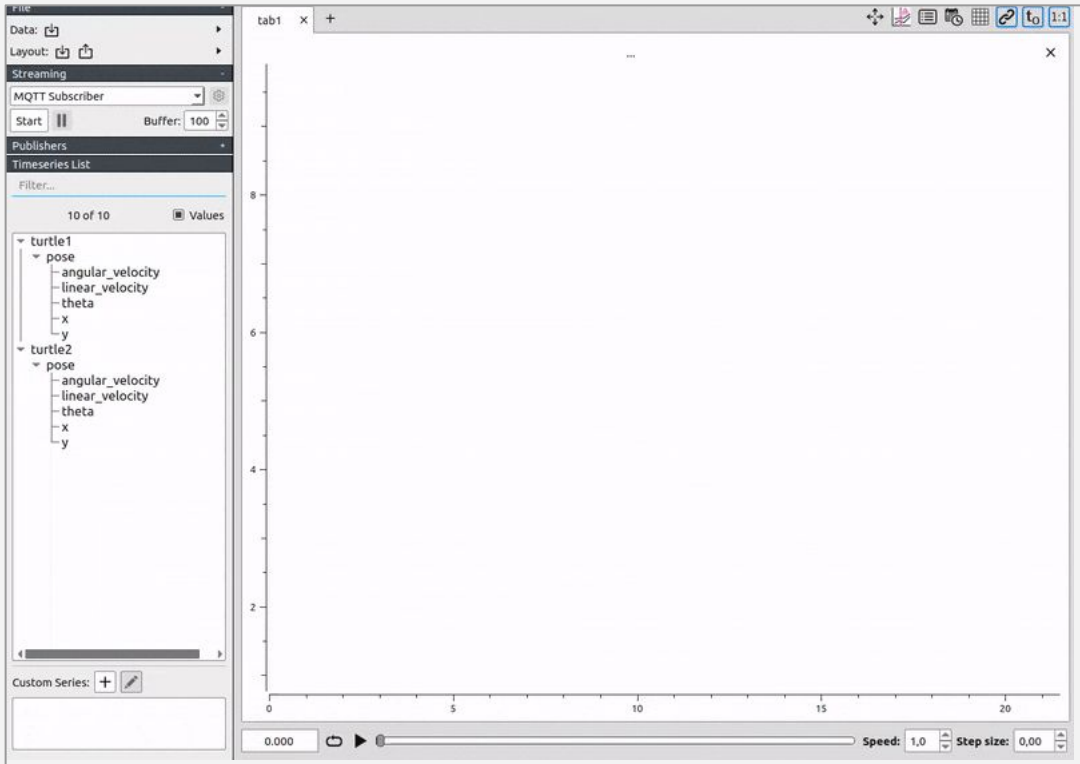
Выбор типа визуализации (сырые  
данные/график/изображение)

# ПРОСМОТР СОДЕРЖИМОГО

<https://github.com/facontidavide/PlotJuggler>

Пакет `plotjuggler`

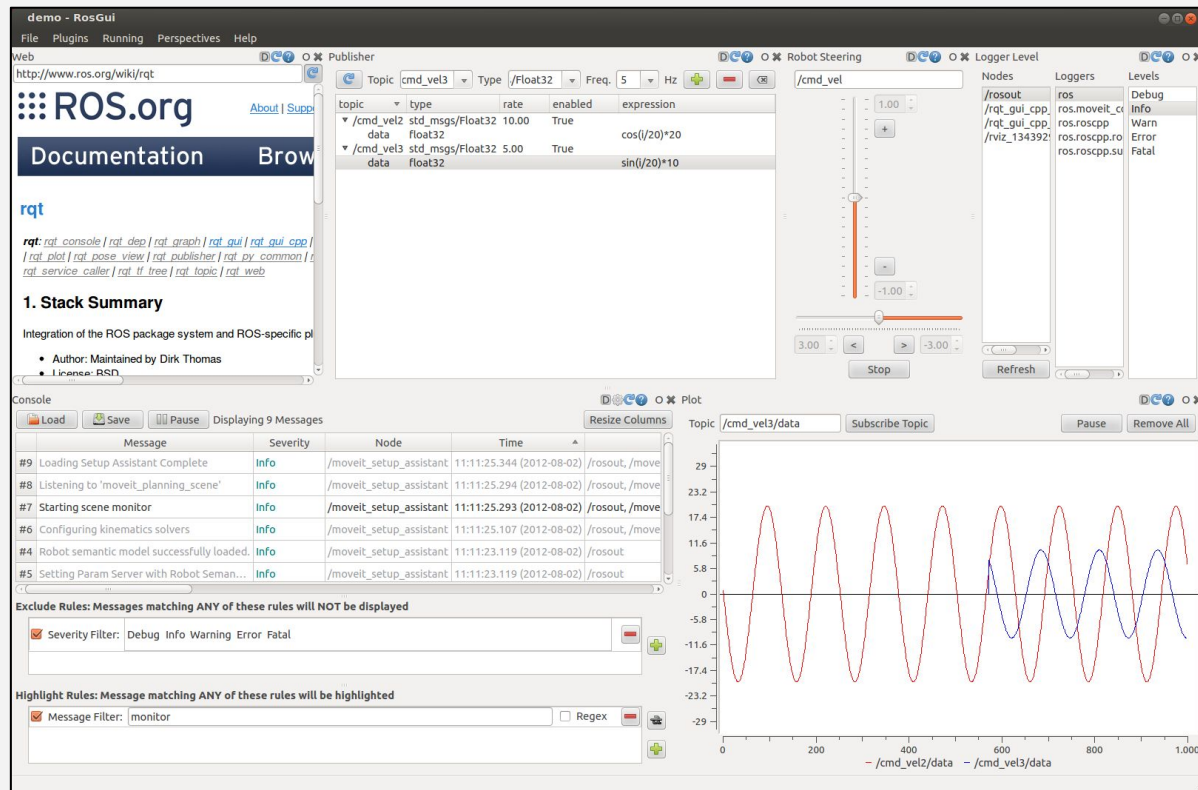
- ❑ `rqt_bag` на максималках
- ❑ Позволяет



# RQT

<http://wiki.ros.org/rqt>

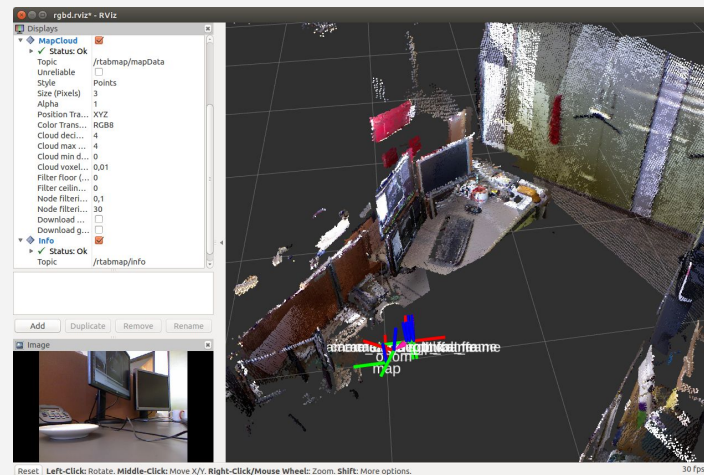
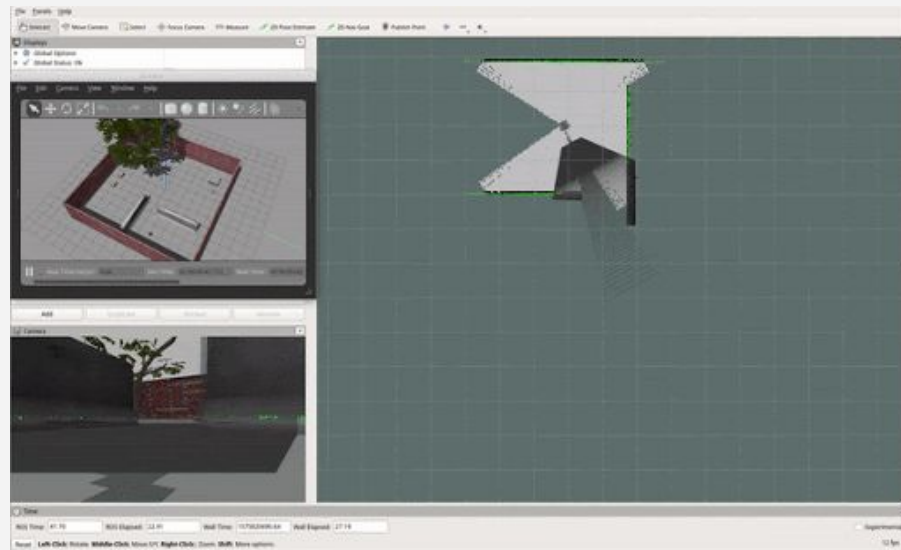
- ❑ Фреймворк для создания GUI ROS-приложений на основе Qt
- ❑ Содержит набор готовых плагинов
- ❑ Предоставляет API для написания своих плагинов



# RVIZ

<http://wiki.ros.org/rviz>

- ❑ Инструмент 2D/3D визуализации в ROS
- ❑ Поддерживает визуализацию распространенных типов данных (карты занятости, лазерные сканы, облака точек, системы координат, траектория и др.), а также отрисовку простых геометрических примитивов (кубы, цилиндры, точки, линии и др.) и даже полноценных CAD-моделей
- ❑ Функционал может быть расширен пользовательскими плагинами

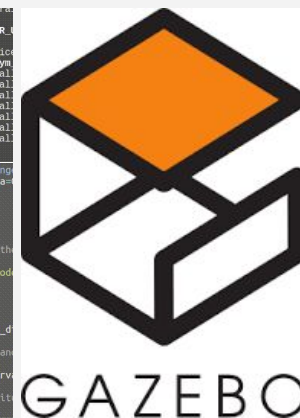
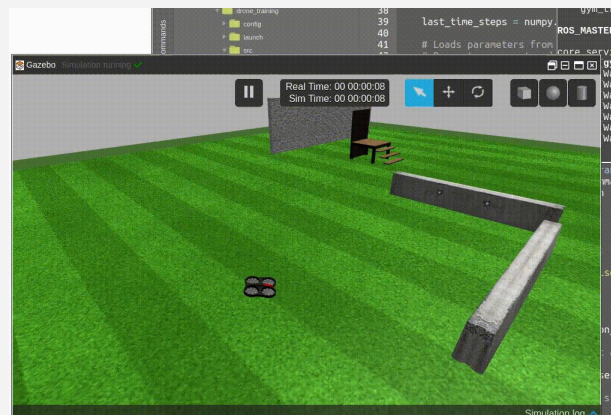
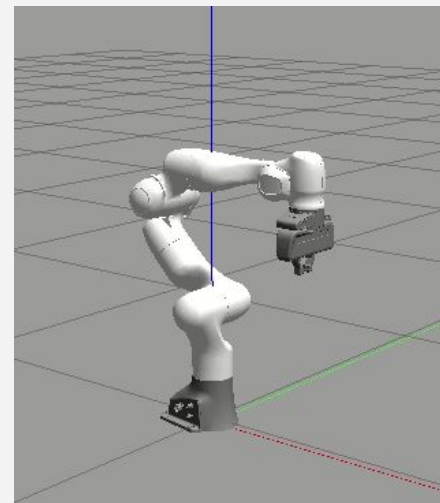
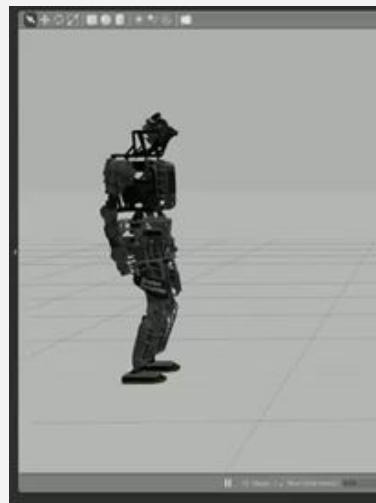




# GAZEBO

[http://wiki.ros.org/gazebo\\_ros\\_pkgs](http://wiki.ros.org/gazebo_ros_pkgs)

- ❑ **Gazebo** — 3D симулятор твердых тел с открытым исходным кодом.
- ❑ Часто применяется в связке с ROS для моделирования роботов и используется для проведения робототехнических соревнований .
- ❑ Gazebo может:
  - ❑ Использовать различные физические движки: ODE, Bullet, и др.
  - ❑ Осуществлять реалистичный рендеринг включая различные источники освещения, тени, текстуры и т.д.
  - ❑ Моделировать сенсоры, включая их шумы измерения: LIDAR, камеры, камеры глубины

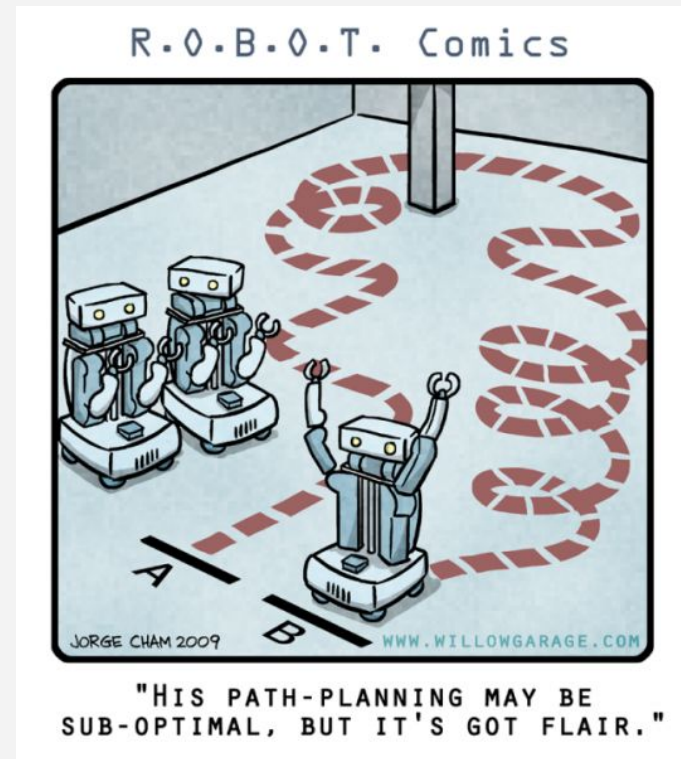




# ROS NAVIGATION STACK

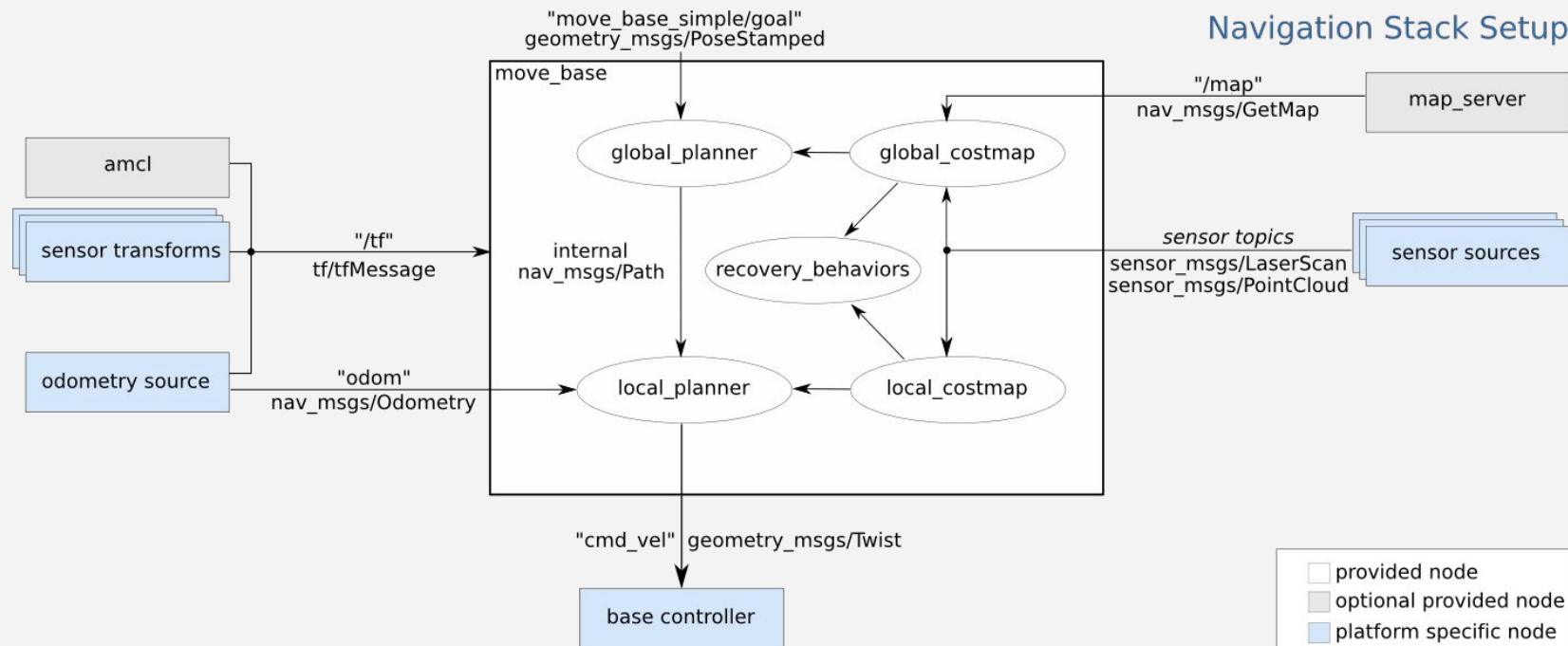
<http://wiki.ros.org/navigation>

- ❑ Стек 2D навигации принимает в качестве входов **одометрию**, **сенсорные данные**, и **целевое положение** и рассчитывает **управление по скорости** (линейной и угловой), позволяющее роботу достигнуть заданной точки.
- ❑ Ограничения стека навигации:
  - ❑ Предназначен для роботов **с дифференциальным приводом** или **голономных** роботов
  - ❑ Робот должен иметь планарный лазерный дальномер (или другой сенсор способный генерировать 2D сканы) для генерации карты и локализации
  - ❑ Подходит для роботов с квадратной/круглой базой. Для роботов других форм планирование пути может быть субоптимальным



# ROS NAVIGATION STACK

<http://wiki.ros.org/navigation>



# TURTLEBOT

<http://wiki.ros.org/Robots/TurtleBot>

<http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/#turtlebot>

## Original TurtleBot

(Discontinued)



---

## TurtleBot 2 Family



TurtleBot 2



TurtleBot 2i



TurtleBot 2e

## TurtleBot 3 Family

Burger



Waffle



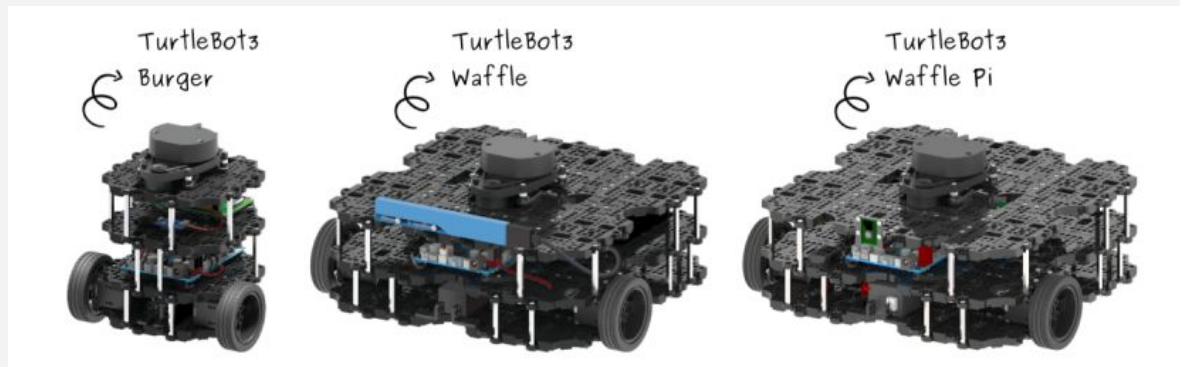
Waffle Pi



# TURTLEBOT SIMULATION. ПОДГОТОВКА

<http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/#turtlebot>

- ❑ `sudo apt update`
- ❑ `sudo apt install ros-melodic-turtlebot3`
- ❑ `cd /root/my_ros_ws/src`
- ❑ `git clone https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git`
- ❑ `source /opt/ros/melodic/setup.zsh`
- ❑ `cd /root/my_ros_ws && catkin_make`
- ❑ `source ./devel/setup.zsh`



# TURTLEBOT SIMULATION. ЗАПУСК

<http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/#turtlebot>

`sudo apt update`

В первом терминале:

```
export TURTLEBOT3_MODEL=${TB3_MODEL}
```

```
    ${TB3_MODEL}: burger, waffle,  
                  waffle_pi
```

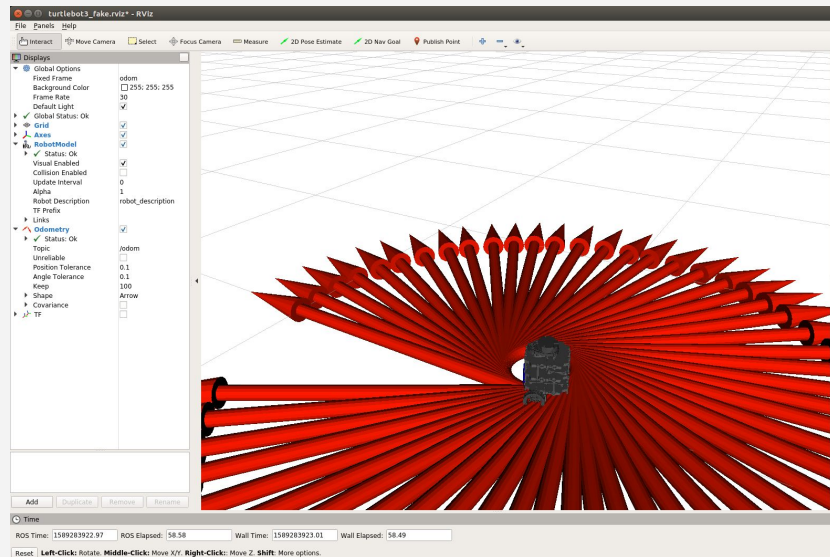
```
roslaunch turtlebot3_fake  
turtlebot3_fake.launch
```

В другом терминале:

```
export TURTLEBOT3_MODEL=${TB3_MODEL}
```

```
    ${TB3_MODEL}: burger, waffle,  
                  waffle_pi
```

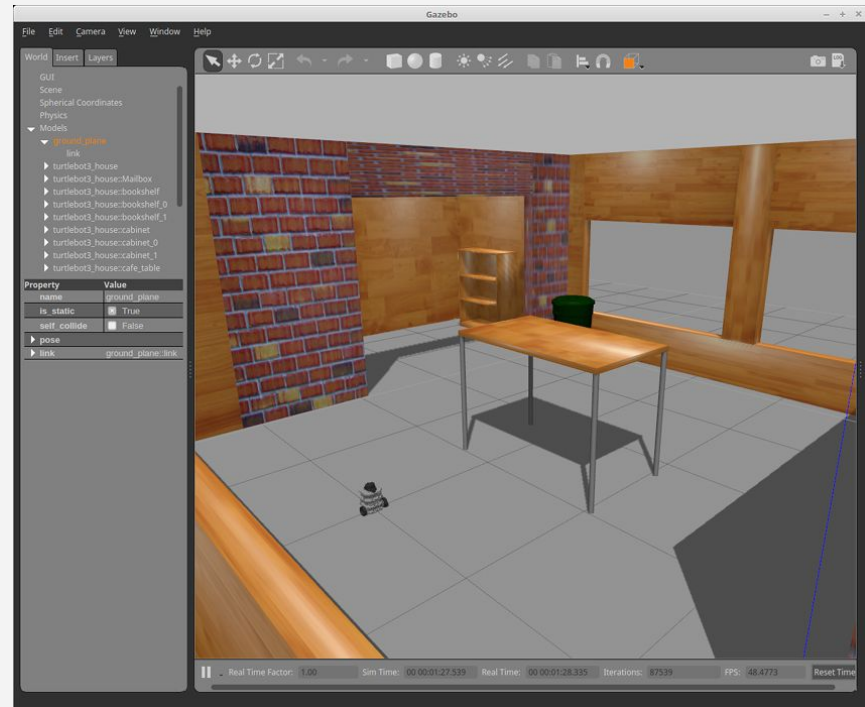
```
roslaunch turtlebot3_teleop  
turtlebot3_teleop_key.launch
```



# TURTLEBOT SIMULATION. COLLISION AVOIDANCE

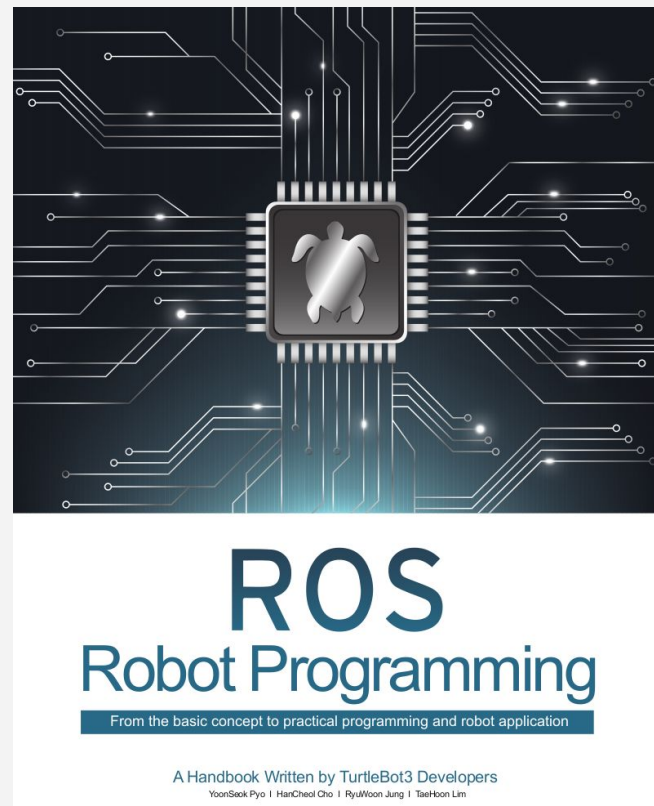
<http://emanual.robotis.com/docs/en/platform/turtlebot3/overview/#turtlebot>

- ❑ В первом терминале:
  - ❑ `export TURTLEBOT3_MODEL=waffle_pi`
  - ❑ `roslaunch turtlebot3_gazebo turtlebot3_house.launch`
- ❑ В другом терминале:
  - ❑ `export TURTLEBOT3_MODEL=waffle_pi`
  - ❑ `roslaunch turtlebot3_gazebo turtlebot3_simulation.launch`
- ❑ В третьем:
  - ❑ `export TURTLEBOT3_MODEL=waffle_pi`
  - ❑ `roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch`



# ДОПОЛНИТЕЛЬНЫЕ ИСТОЧНИКИ

1. Книга: ROS Robot Programming.  
YoonSeok Pyo, HanCheol Cho, RyuWoon Jung, TaeHoon Lim (Eng)
2. Обучающие инструкции ROS (Eng)
3. Введение в ROS от Voltbro (Rus)
4. Clearpath Robotics ROS Tutorial (Eng)



# ИНФОРМАЦИЯ О ПРЕЗЕНТАЦИИ

Эта презентация была подготовлена Олегом Шипитько в рамках курса “Моделирование колесных роботов” кафедры когнитивных технологий Московского физико-технического института (МФТИ). Автор выражает благодарность, авторам, чьи материалы были использованы в презентации. В случае, если вы обнаружили в презентации свои материалы, свяжитесь со мной, для включения в список авторов заимствованных материалов.

This presentation was prepared by Oleg Shipitko as part of the “Mobile Robotics” course at the Department of Cognitive Technologies, Moscow Institute of Physics and Technology. The author is grateful to the authors whose materials were used in the presentation. If you find your materials in a presentation, contact me to be included in the list of contributing authors.