From Simulink Models to ROS 2 Control
- Streamlining Robotic Controller Development

Josh Chen

Senior Software Engineer,
ROS Toolbox
MathWorks

Gijs van der Hoorn

Researcher,
Robot Dynamics Group
TU Delft

YJ Lim

Principal Product Manager of
Robotics
MathWorks

MATLAB® & SIMULINK®

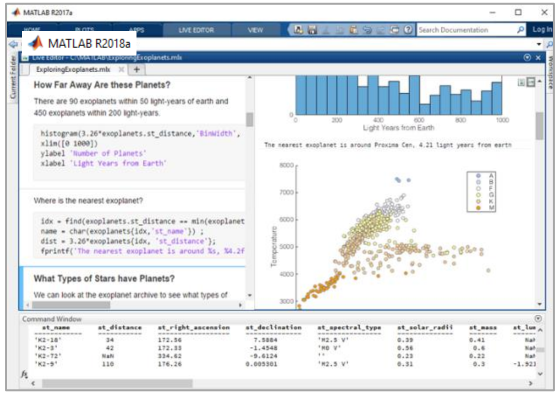
Our Products

MATLAB®, the language of engineers and scientists, is a programming environment for algorithm development, data analysis, visualization, and numeric computation.

Simulink® is a graphical programming environment for modeling, simulation, and analyzing dynamical systems. Control development tool.

More than 100 add-on toolboxes for specialized tasks

MATLAB



How Far Away Are these Planets?

There are 90 exoplanets within 50 light-years of earth and 450 exoplanets within 200 light-years.

```
histogram(3.26*exoplanets.st_distance,'BinWidth',
xlim([0 1000])
ylabel 'Number of Planets'
xlabel 'Light Years from Earth'
```

```
The nearest exoplanet is around Proxima Cen, 4.21 light years from earth
```

Where is the nearest exoplanet?

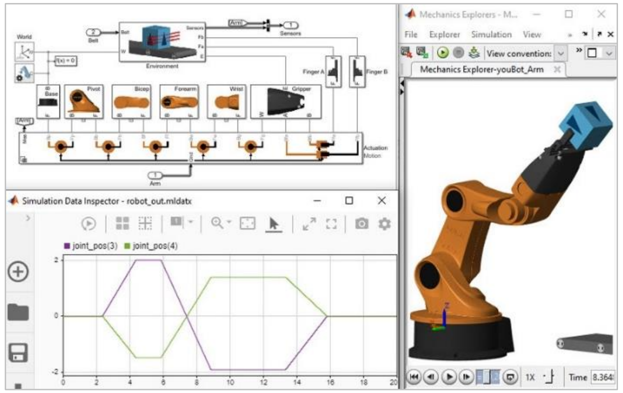
```
idx = find(exoplanets.st_distance == min(exoplanet
name = char(exoplanets{idx,'st_name'}) ;
dist = 3.26*exoplanets{idx, 'st_distance'};
fprintf('The nearest exoplanet is around %s, %4.2f
```

What Types of Stars have Planets?

We can look at the exoplanet archive to see what types of

```
   st_name     st_distance   st_right_ascension   st_declination   st_spectral_type   st_solar_radii   st_mass   st_lum
  ---------    -----------   ------------------   --------------   ----------------   --------------   -------   ------
  'K2-18'           34             172.56              7.5884          'M2.5 V'             0.39          0.41      NaN
  'K2-3'            42             172.33             -1.4548          'M0 V'               0.56          0.6       NaN
  'K2-72'          NaN             334.62             -9.6124          ''                   0.23          0.22      NaN
  'K2-9'           110             176.26              0.005301        'M2.5 V'             0.31          0.3     -1.921
```

Simulink



Robotics System Toolbox
Design, simulate, test, and deploy robotics applications

Deep Learning Toolbox
Design, train, and analyze deep learning networks

ROS Toolbox
Design, simulate, and deploy ROS-based applications

MathWorks Supports Robotics and Autonomous Systems

Design, Simulate, Test, and Deploy

Manipulators / Cobots

Mobile Robots and Ground Vehicles

Marine Robots

ROS 2

ROS Toolbox

Design, simulate, and deploy ROS-based applications

Get a free trial

View pricing

Have questions? Contact Sales.

Agenda

Introduction

- Program ROS and ROS 2 using MATLAB and Simulink

From Simulink to ros2_control generation

- Automated ros2_control plugin

Summary

ROS programming can be done in either Python or C++

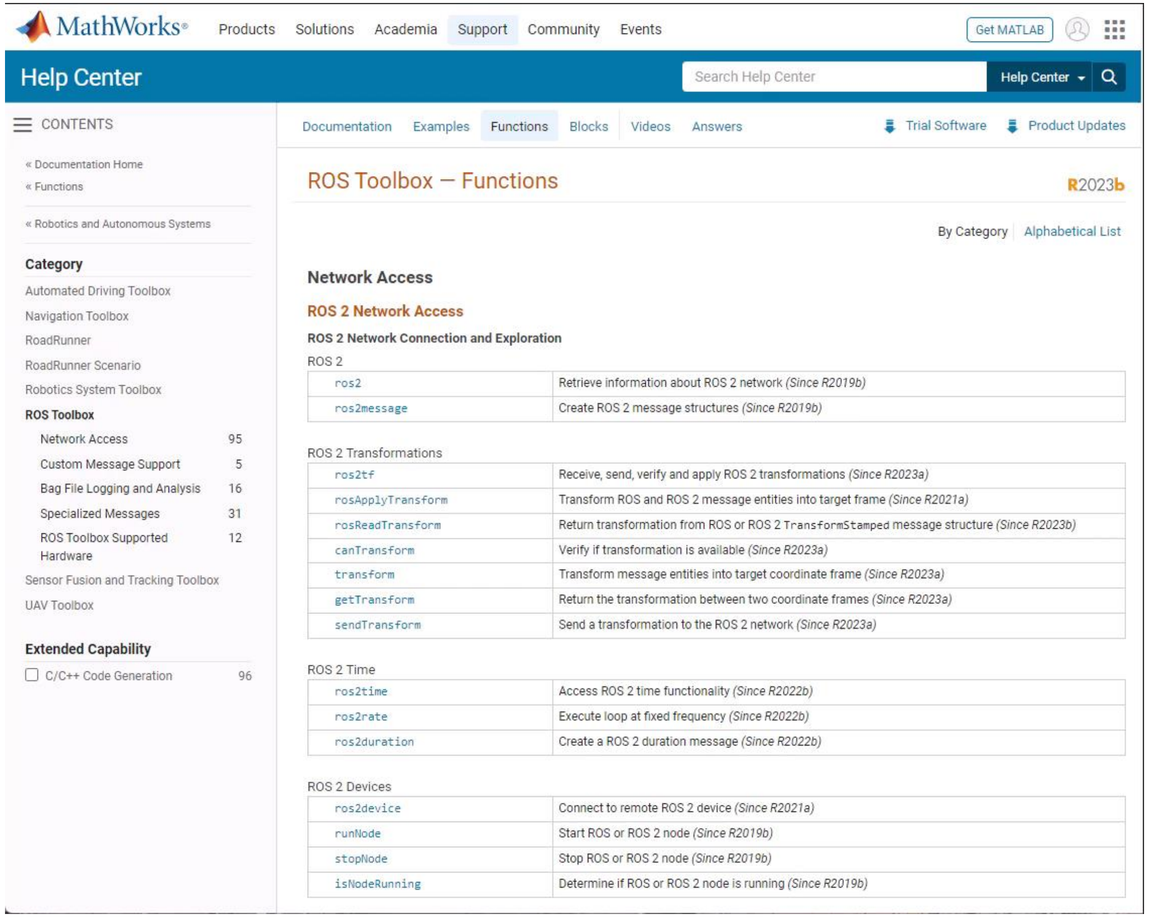
Java

and you can also do in

MATLAB/Simulink!!

MATLAB/Simulink

How to do ROS programming using MATLAB?



« Documentation Home
« Functions
« Robotics and Autonomous Systems

Category

Automated Driving Toolbox	
Navigation Toolbox	
RoadRunner	
RoadRunner Scenario	
Robotics System Toolbox	
ROS Toolbox	
Network Access	95
Custom Message Support	5
Bag File Logging and Analysis	16
Specialized Messages	31
ROS Toolbox Supported Hardware	12
Sensor Fusion and Tracking Toolbox	
UAV Toolbox	

Extended Capability

☐ C/C++ Code Generation 96

ROS Toolbox — Functions

R2023b

By Category | Alphabetical List

Network Access

ROS 2 Network Access

ROS 2 Network Connection and Exploration

ROS 2

`ros2`	Retrieve information about ROS 2 network *(Since R2019b)*
`ros2message`	Create ROS 2 message structures *(Since R2019b)*

ROS 2 Transformations

`ros2tf`	Receive, send, verify and apply ROS 2 transformations *(Since R2023a)*
`rosApplyTransform`	Transform ROS and ROS 2 message entities into target frame *(Since R2021a)*
`rosReadTransform`	Return transformation from ROS or ROS 2 `TransformStamped` message structure *(Since R2023b)*
`canTransform`	Verify if transformation is available *(Since R2023a)*
`transform`	Transform message entities into target coordinate frame *(Since R2023a)*
`getTransform`	Return the transformation between two coordinate frames *(Since R2023a)*
`sendTransform`	Send a transformation to the ROS 2 network *(Since R2023a)*

ROS 2 Time

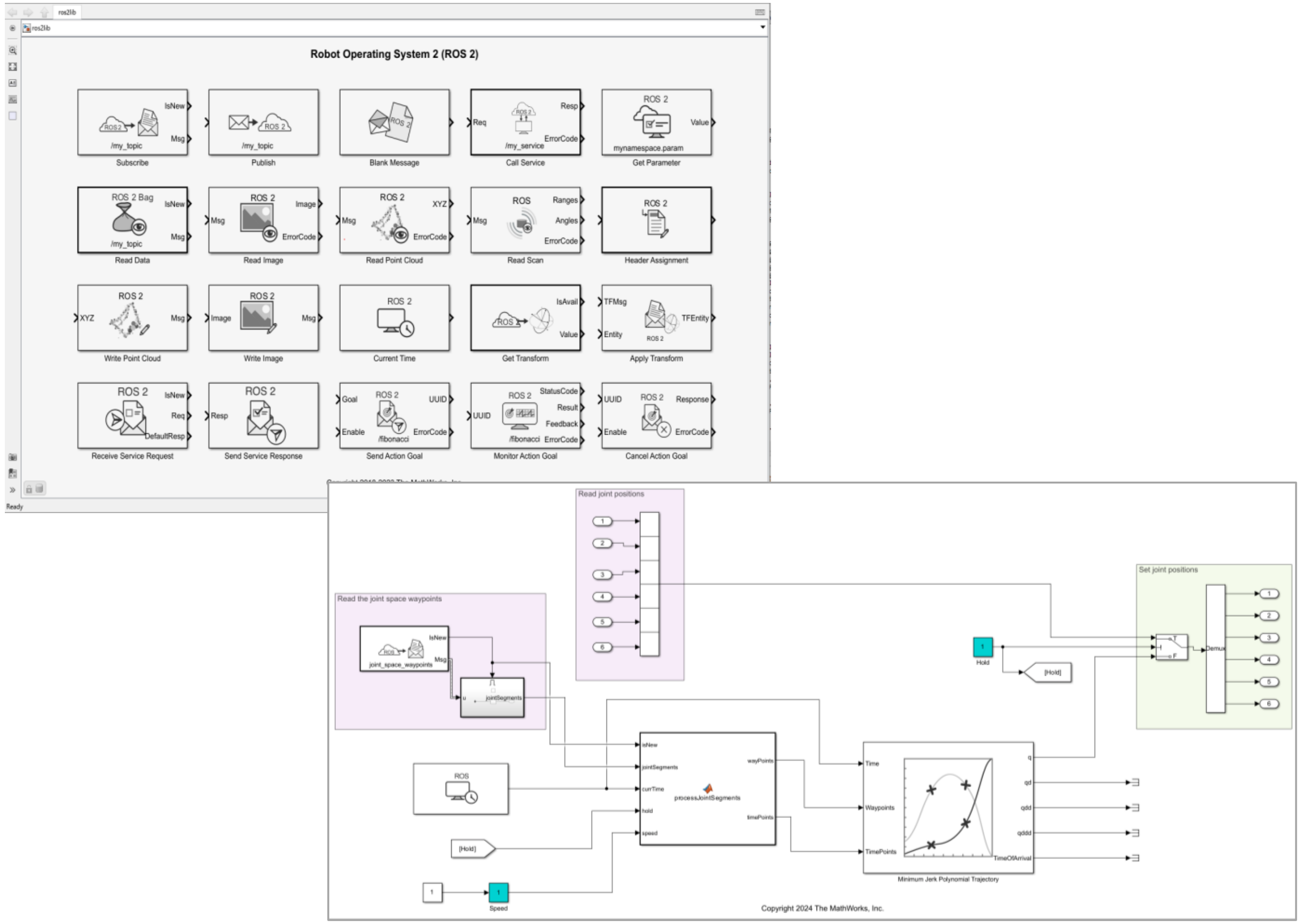
`ros2time`	Access ROS 2 time functionality *(Since R2022b)*
`ros2rate`	Execute loop at fixed frequency *(Since R2022b)*
`ros2duration`	Create a ROS 2 duration message *(Since R2022b)*

ROS 2 Devices

`ros2device`	Connect to remote ROS 2 device *(Since R2021a)*
`runNode`	Start ROS or ROS 2 node *(Since R2019b)*
`stopNode`	Stop ROS or ROS 2 node *(Since R2019b)*
`isNodeRunning`	Determine if ROS or ROS 2 node is running *(Since R2019b)*

- Create ROS nodes, publishers, subscribers, services, and actions directly via MATLAB APIs.
- Enhance ROS programming with MATLAB's Toolboxes, e.g., Navigation Toolbox and Computer Vision Toolbox
- No CMAKE or C++ knowledge required.
- Leverage MATLAB/Embedded Coders for automatic C++ and CUDA ROS code generation.

How to do ROS programming using Simulink?



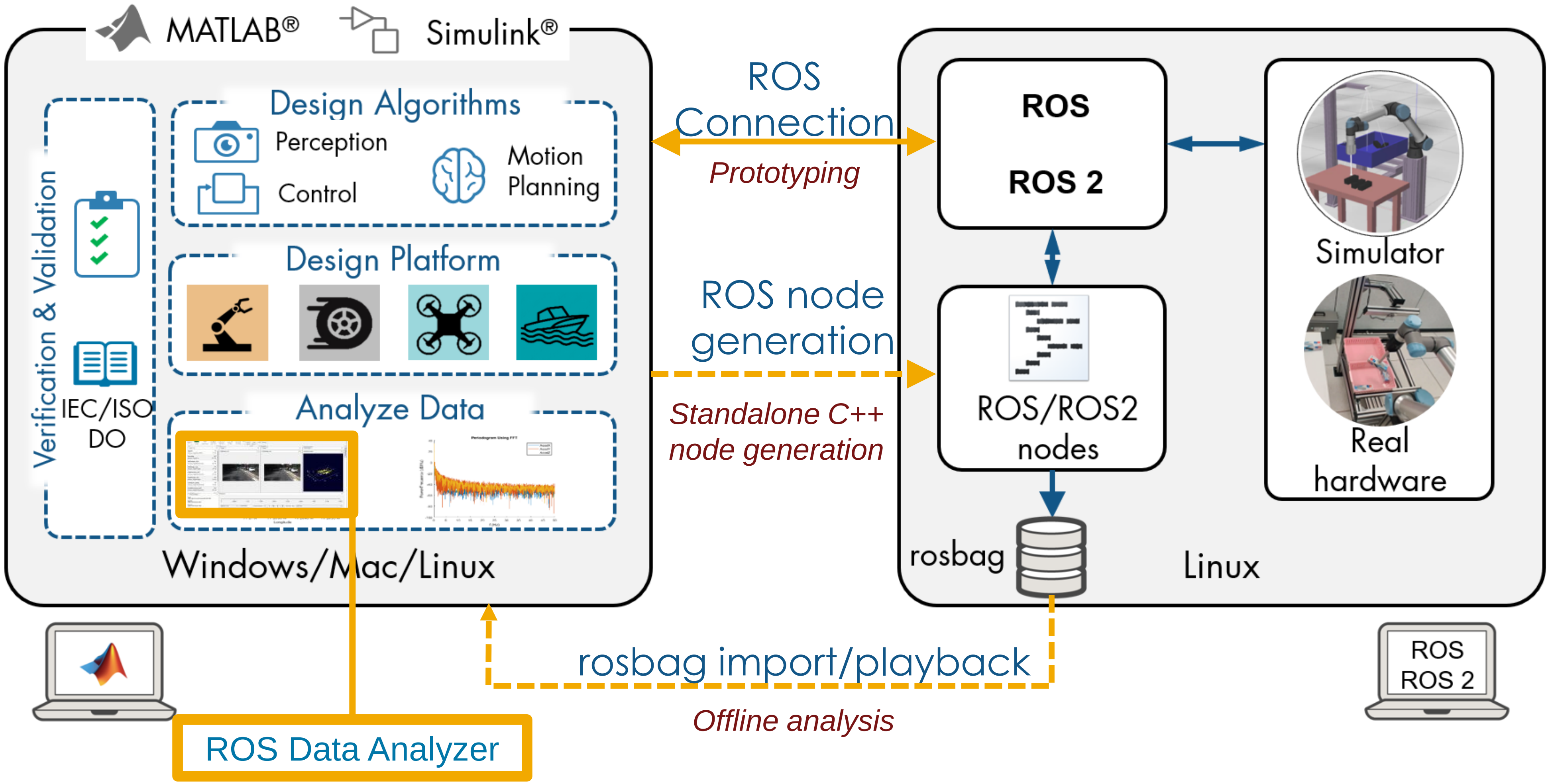
- Utilize Simulink blocks for publishers, subscribers, services, and actions.
- Modeling and simulation for Model-Based Design with Simulink
- No need for CMAKE or C++ expertise.
- Automatically generate C++ and CUDA ROS code with Simulink/Embedded Coders.

We support major ROS functionalities in both MATLAB and Simulink for ROS and ROS2

	ROS	2
	✓ • Topic – Publish / Subscribe • Service server, Service client • Action – Client / Server • Parameter – Get / Set • ROS TF • Custom messages • rosbag reader, rosbag writer • Code generation • CUDA ROS code generation	✓ • Topic – Publish / Subscribe • Service server, Service client • Action – Client / Server • Parameter – Get / Set • ROS2 TF • Custom messages • ros2bag reader, ros2bag writer • Code generation • CUDA ROS2 code generation
	✓ • Topic – Publish / Subscribe • Service – Call • Parameter – Get / Set • ROS Time • rosbag playback / record • Code generation (Local/Remote) • CUDA ROS code generation	✓ • Topic – Publish / Subscribe • Service – Call / Server • Action – Client • Parameter – Get • ROS2 Time • ros2bag playback / Record • Code generation(Local/Remote) • CUDA ROS2 code generation
ROS Distro	• ROS Noetic	• ROS2 Humble **Switchable DDS**

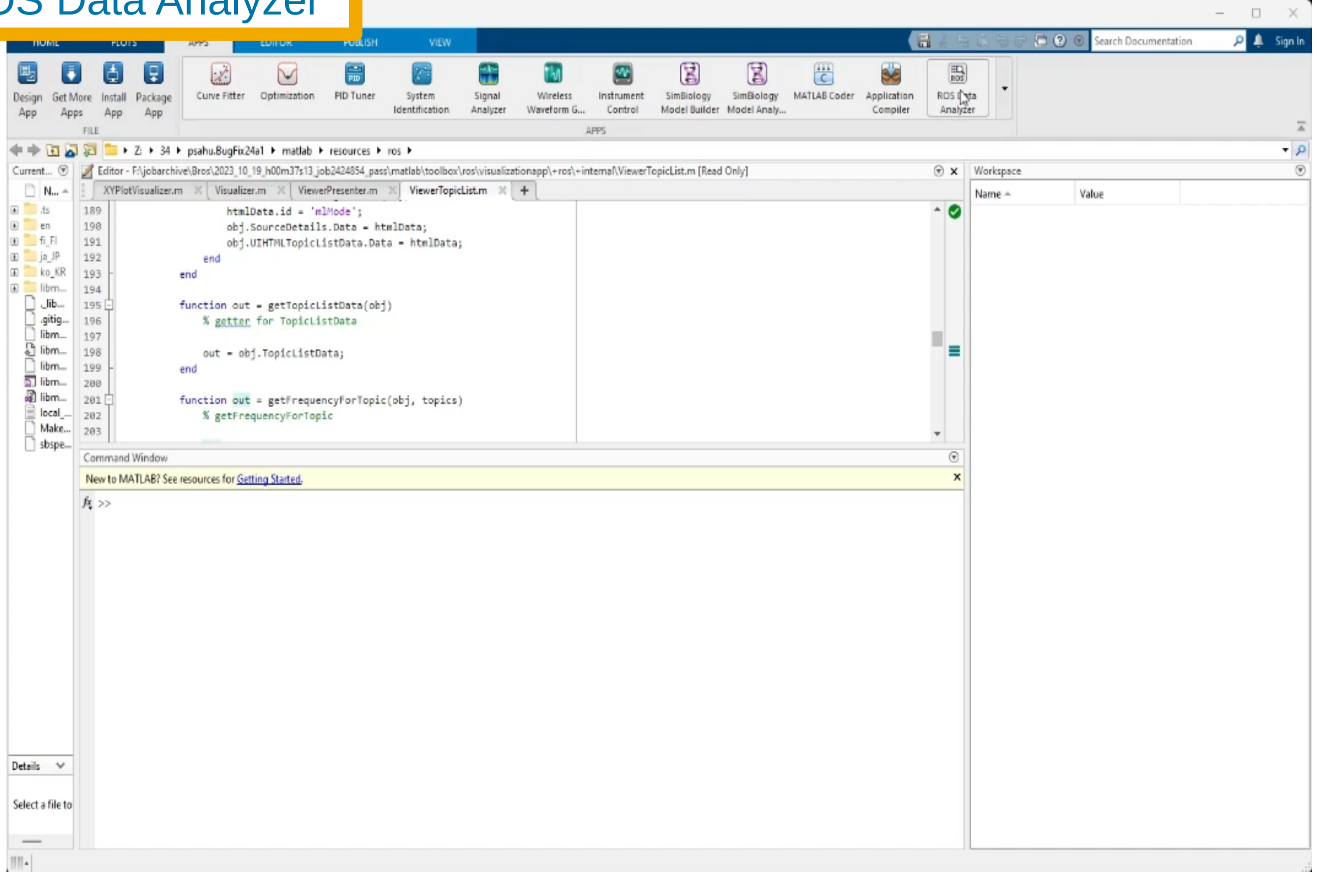
+ **Ease of Use Tools**

MATLAB and Simulink Simplify ROS and ROS 2 Programming



Visualize and analyze ros(2)bag and live ROS data

ROS Data Analyzer



Editor - F:\jobarchive\Bros\2023_10_19_h00m37s13_job2424854_pass\matlab\toolbox\ros\visualizationapp\+ros\+internal\ViewerTopicList.m [Read Only]

```
            htmlData.id = 'mlMode';
            obj.SourceDetails.Data = htmlData;
            obj.UIHTMLTopicListData.Data = htmlData;
        end
    end

    function out = getTopicListData(obj)
        % getter for TopicListData

        out = obj.TopicListData;
    end

    function out = getFrequencyForTopic(obj, topics)
        % getFrequencyForTopic
```

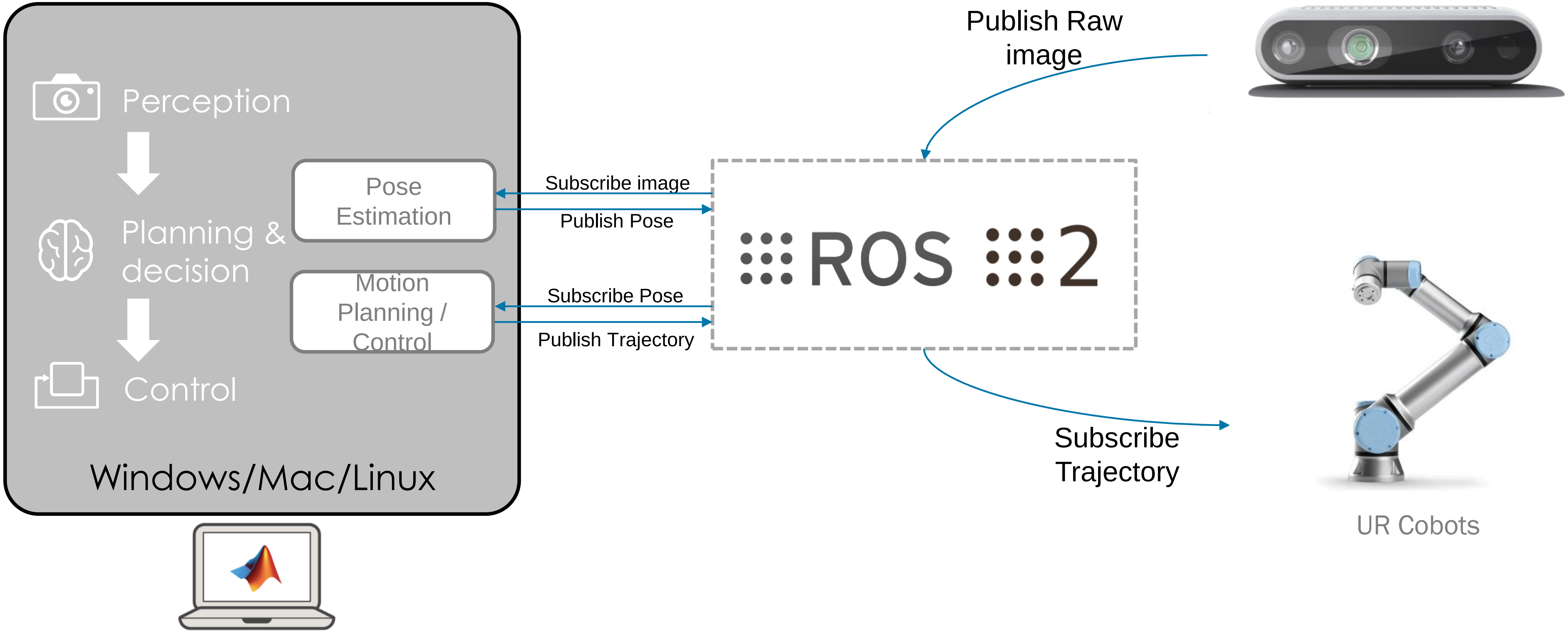
New to MATLAB? See resources for Getting Started.

```
fx >>
```

- Visualize both ROS 1 and ROS2 bag files
- Visualize the live ROS data
- Use tags and bookmarks to ros(2)bag
- Read and visualize ros(2)bags stored in AWS S3

Please come over to the **MathWorks** table during the Lab Tours/Demo Session in the afternoon!!!

MATLAB connects with UR Cobots Via ROS and ROS 2



Perception

Planning & decision

Control

Pose Estimation

Motion Planning / Control

Windows/Mac/Linux

Subscribe image

Publish Pose

Subscribe Pose

Publish Trajectory

ROS 2

Publish Raw image

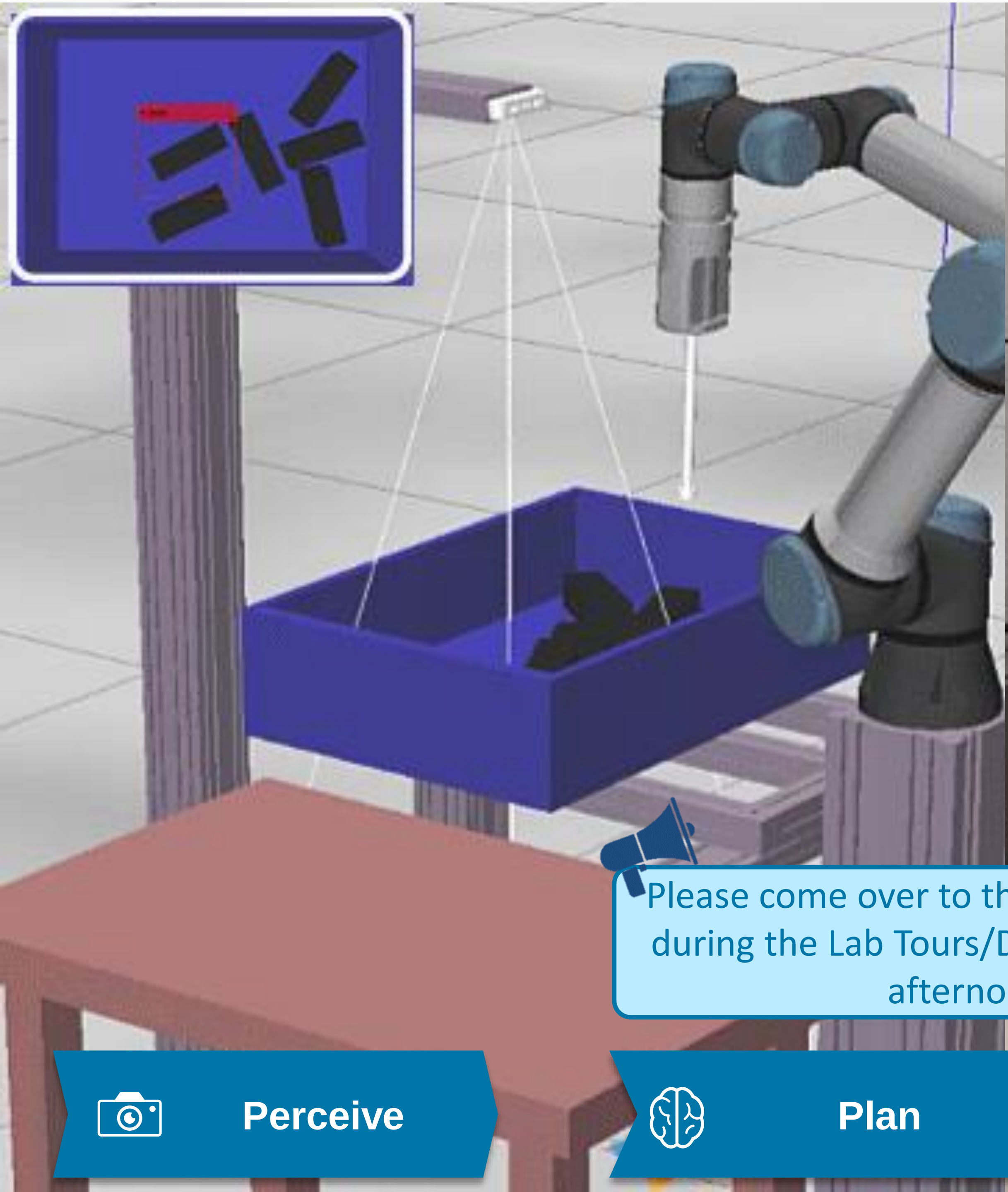
Subscribe Trajectory

UR Cobots

MATLAB

Robotics System Toolbox

ROS Toolbox



Please come over to the **MathWorks** table during the Lab Tours/Demo Session in the afternoon!!!

Perceive

Plan

Control

AI

Agenda

Introduction

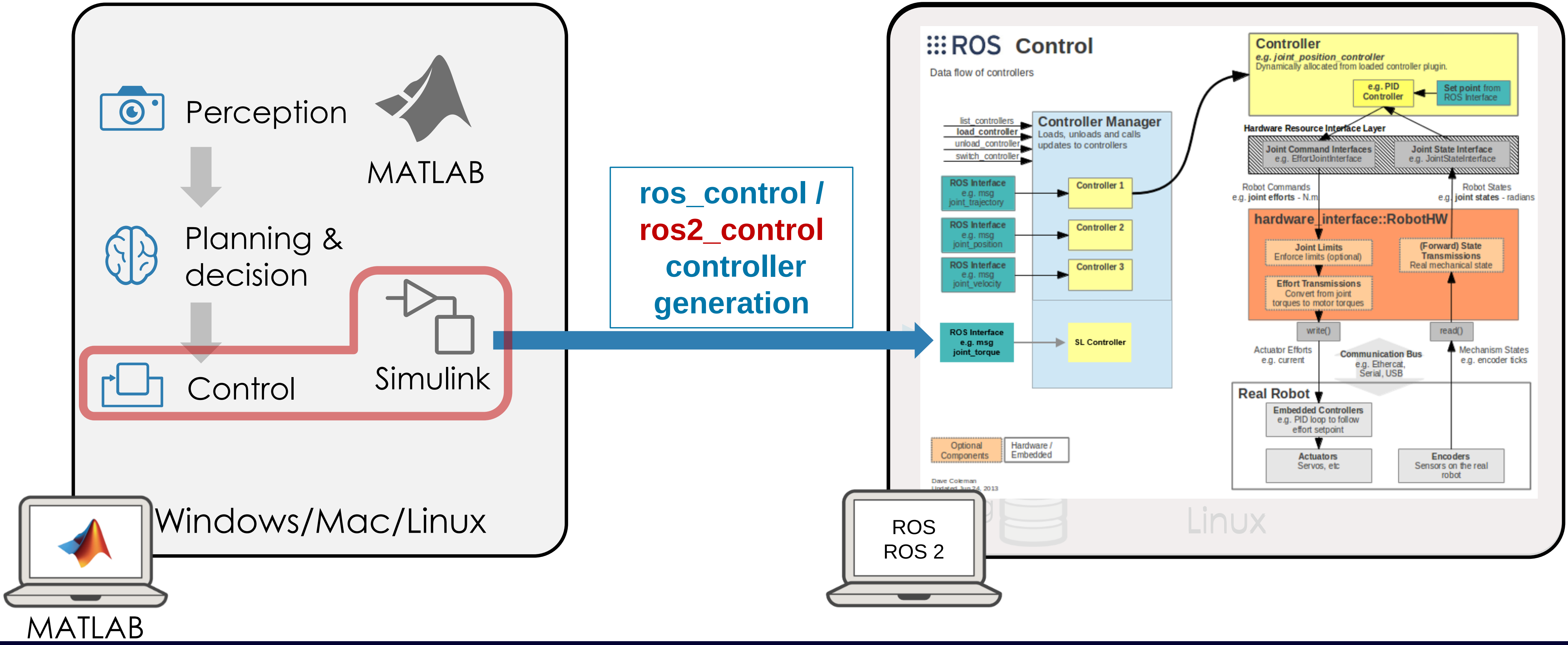
- Program ROS and ROS 2 using MATLAB and Simulink

From Simulink to ros2_control generation

- Automated ros2_control plugin

Summary

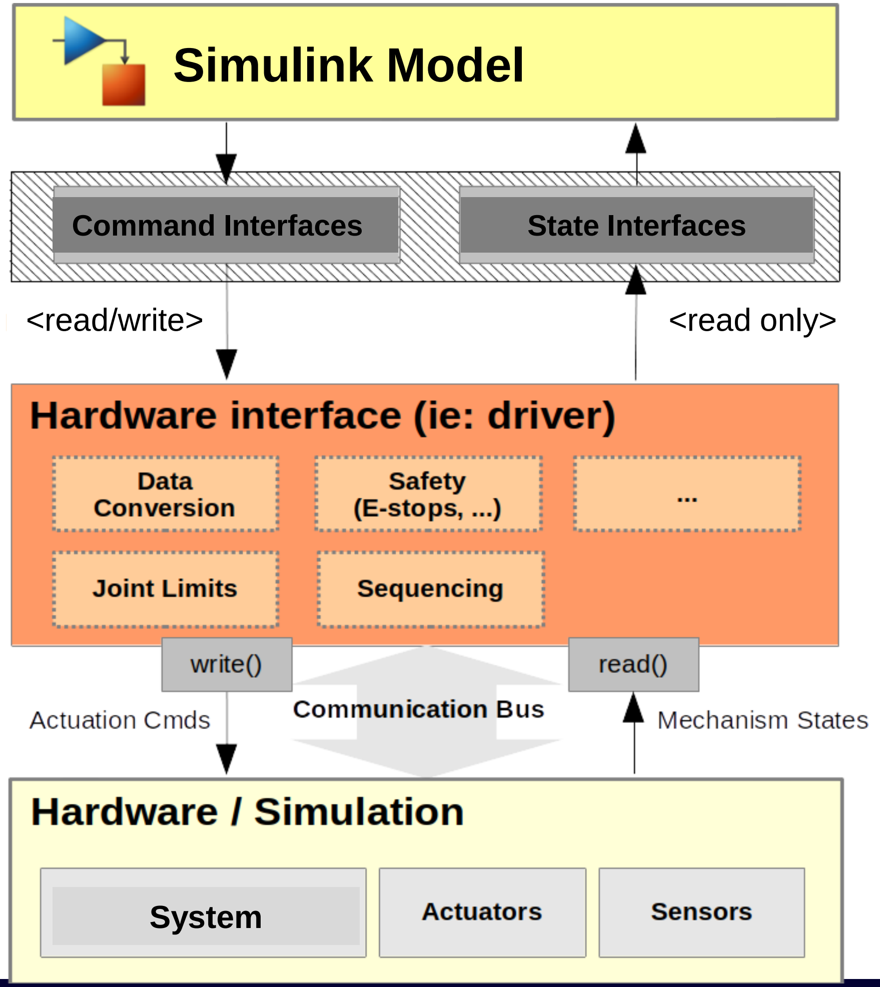
Generate *ros2_control* plugin from Simulink


Perception
MATLAB
Planning & decision
Control
Simulink
Windows/Mac/Linux
MATLAB
ROS Control
Data flow of controllers
list_controllers
load_controller
unload_controller
switch_controller
Controller Manager
Loads, unloads and calls updates to controllers
ROS Interface e.g. msg joint_trajectory
ROS Interface e.g. msg joint_position
ROS Interface e.g. msg joint_velocity
Controller 1
Controller 2
Controller 3
ROS Interface e.g. msg joint_torque
SL Controller
Controller
e.g. joint_position_controller
Dynamically allocated from loaded controller plugin.
e.g. PID Controller
Set point from ROS Interface
Hardware Resource Interface Layer
Joint Command Interfaces e.g. EffortJointInterface
Joint State Interface e.g. JointStateInterface
Robot Commands e.g. joint efforts - N.m
Robot States e.g. joint states - radians
hardware_interface::RobotHW
Joint Limits Enforce limits (optional)
Effort Transmissions Convert from joint torques to motor torques
(Forward) State Transmissions Real mechanical state
write()
read()
Actuator Efforts e.g. current
Communication Bus e.g. Ethercat, Serial, USB
Mechanism States e.g. encoder ticks
Real Robot
Embedded Controllers e.g. PID loop to follow effort setpoint
Actuators Servos, etc
Encoders Sensors on the real robot
Optional Components
Hardware / Embedded
Dave Coleman
Updated Jun 24, 2013
Linux
ROS
ROS 2

ros_control / ros2_control controller generation

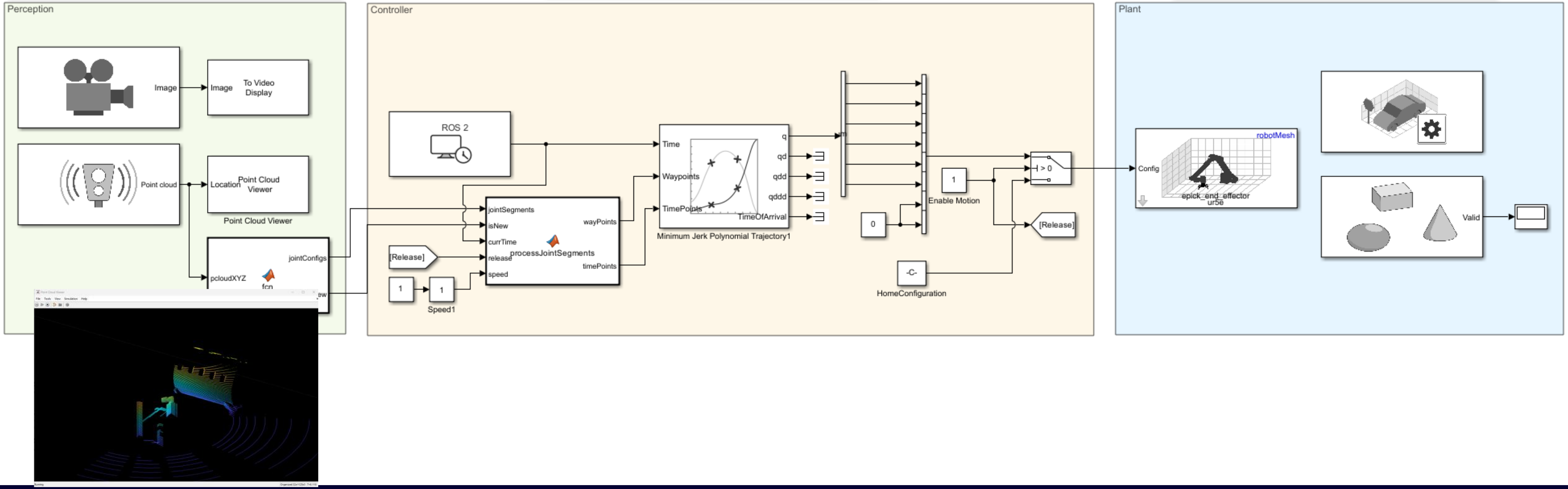
Recap - *ros2_control* architecture

- Layered architecture
- Single process (multi-threaded)
- Determinism *within* node (execution)
- OEM provides up to the *interfaces layer*
- Hardware Interface transforms data:
  - From HW to ROS (ex: enc ticks → rad)
  - From ROS to HW (ex: rad → enc ticks)
- Combine Hardware Interfaces (OEM1, OEM2, …)
- Controllers are user-facing
- Controllers inheriting from Lifecycle nodes

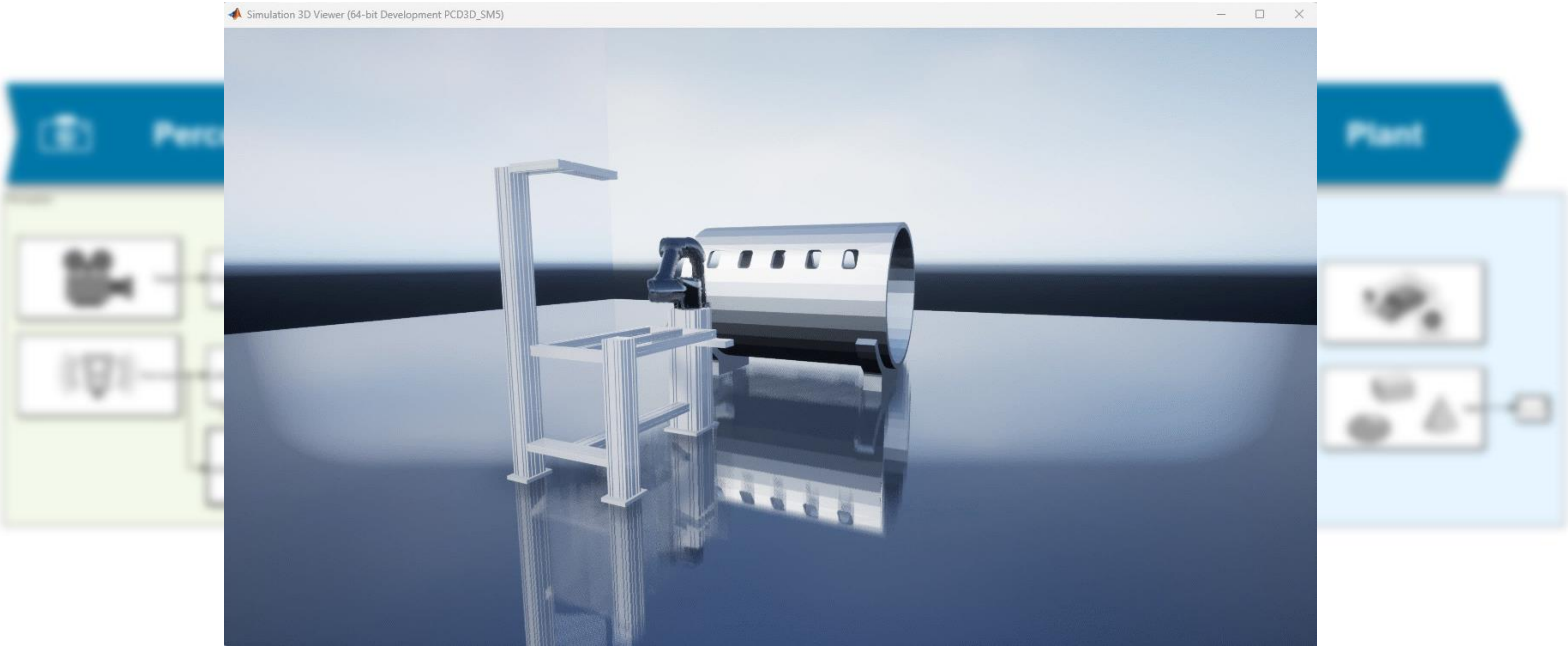


Case Study: Model-Based Design for Painting Robots

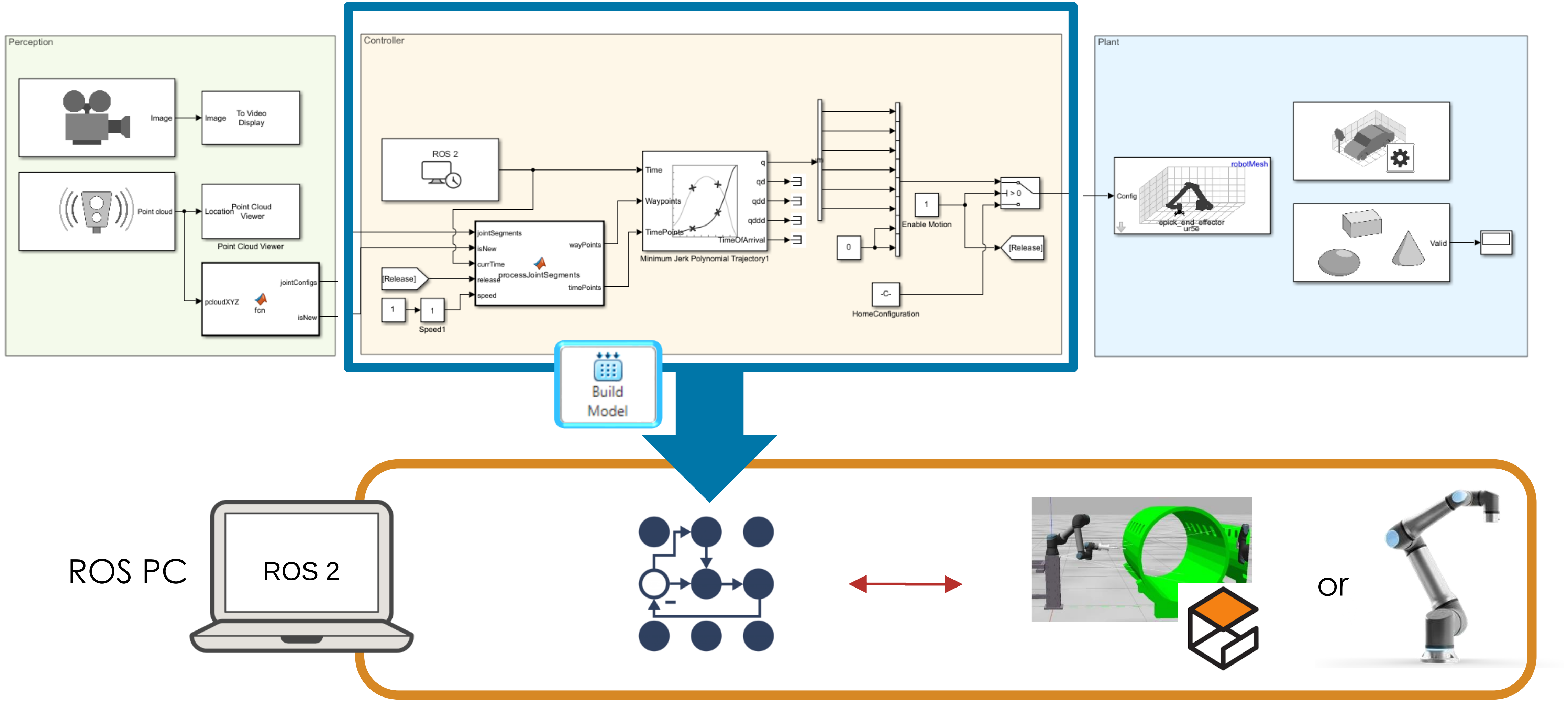
Perceive **Plan / Control** **Plant**



Case Study: Model-Based Design for Painting Robots

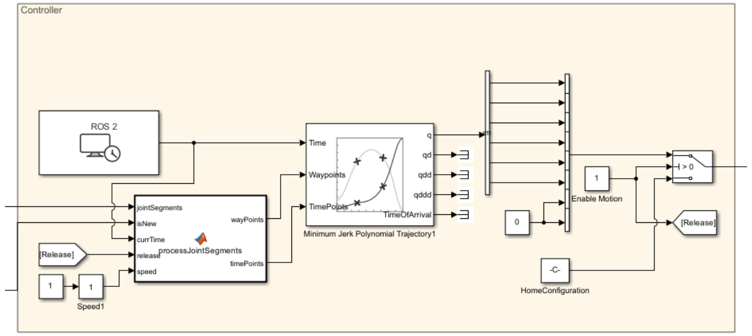


Case Study: Model-Based Design for Painting Robots

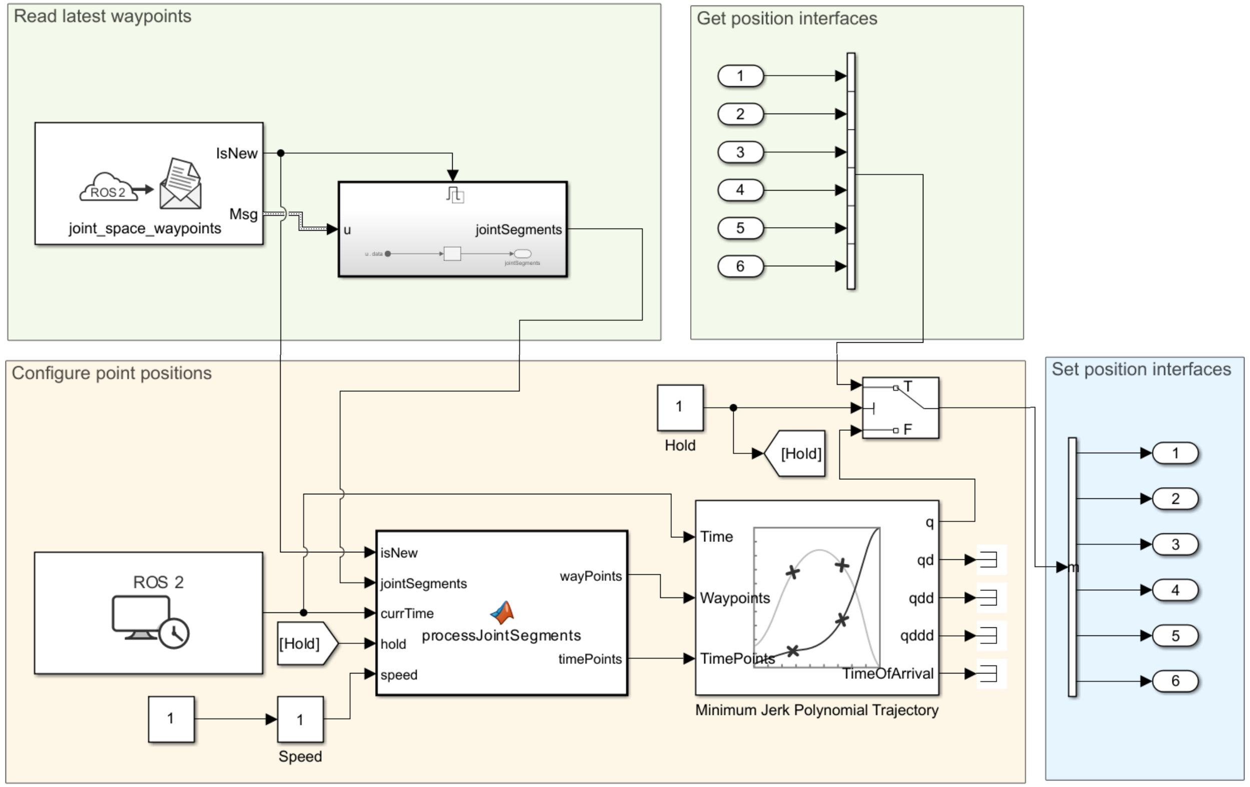


Case Study: Model-Based Design for Painting Robots

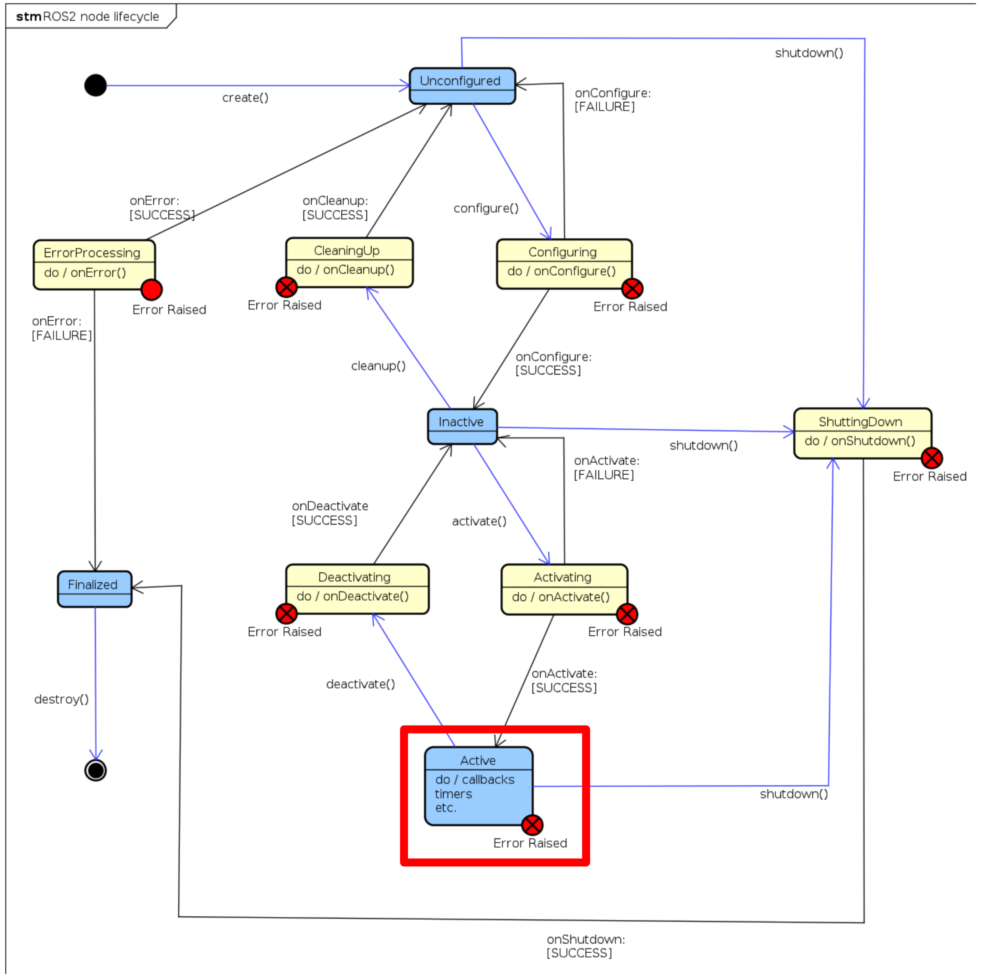
Simulink Control Model

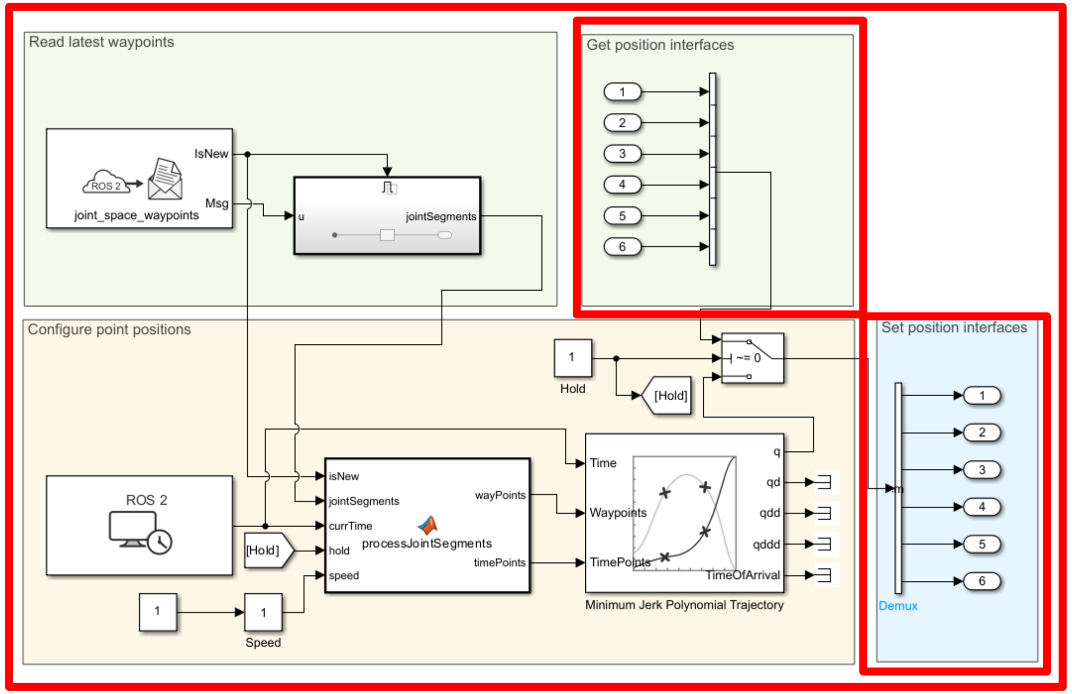


Configure ros2_control interfaces



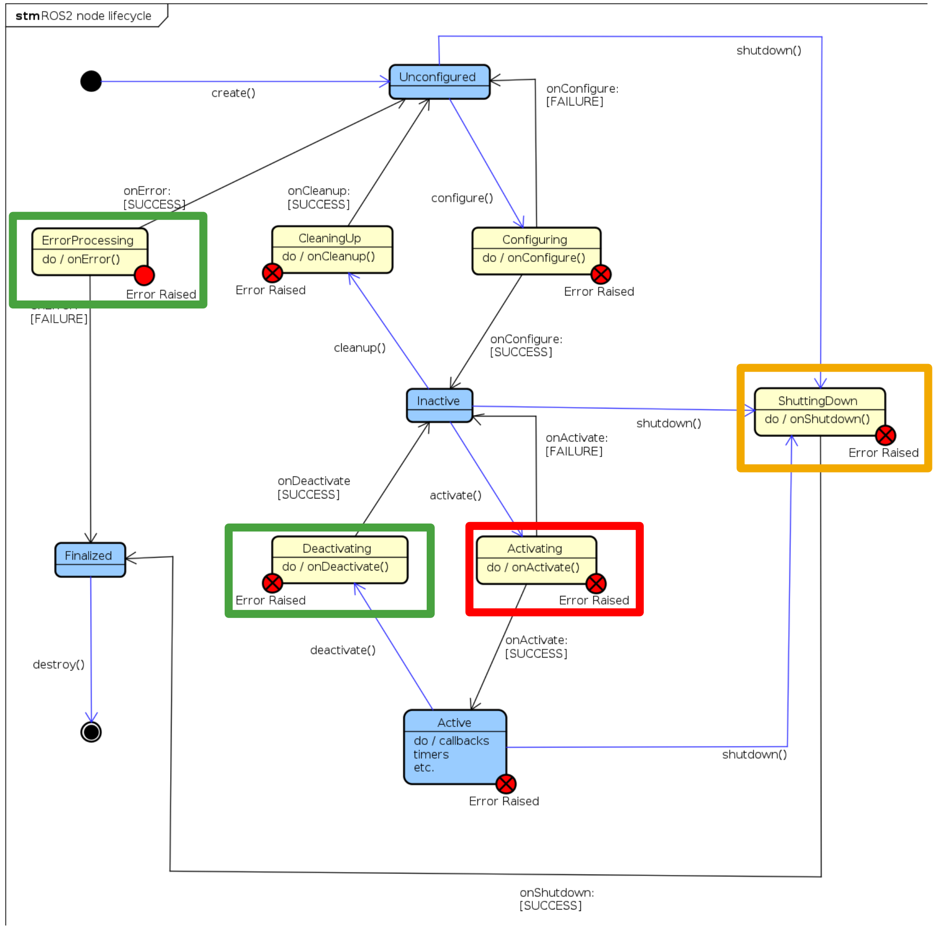
Mapping between Simulink and Lifecycle Node



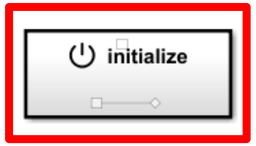


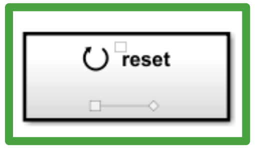
Mapping between Simulink and Lifecycle Node

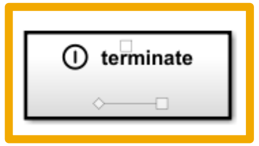
ROS 2 Node Lifecycle



Simulink Code generation Architecture

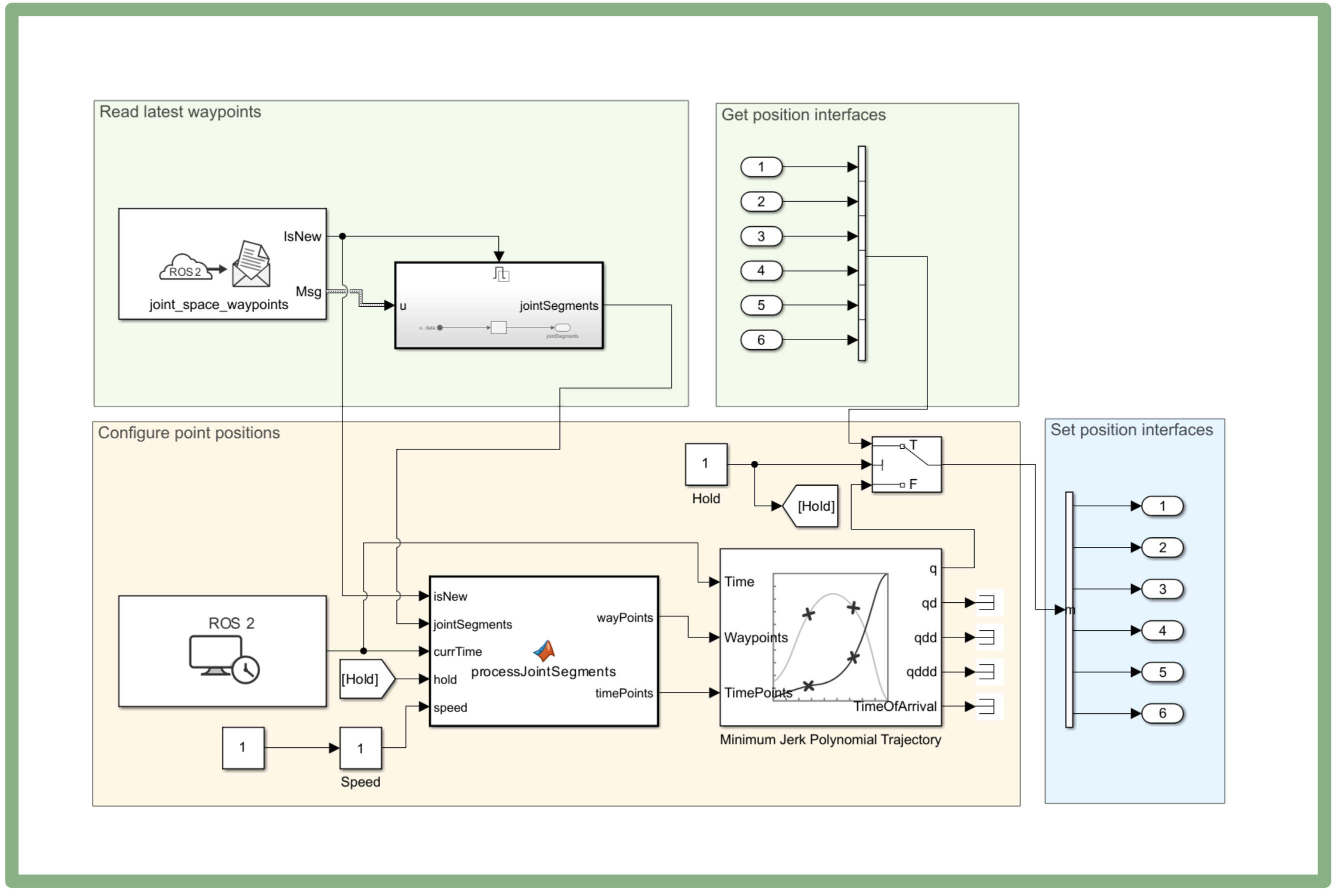


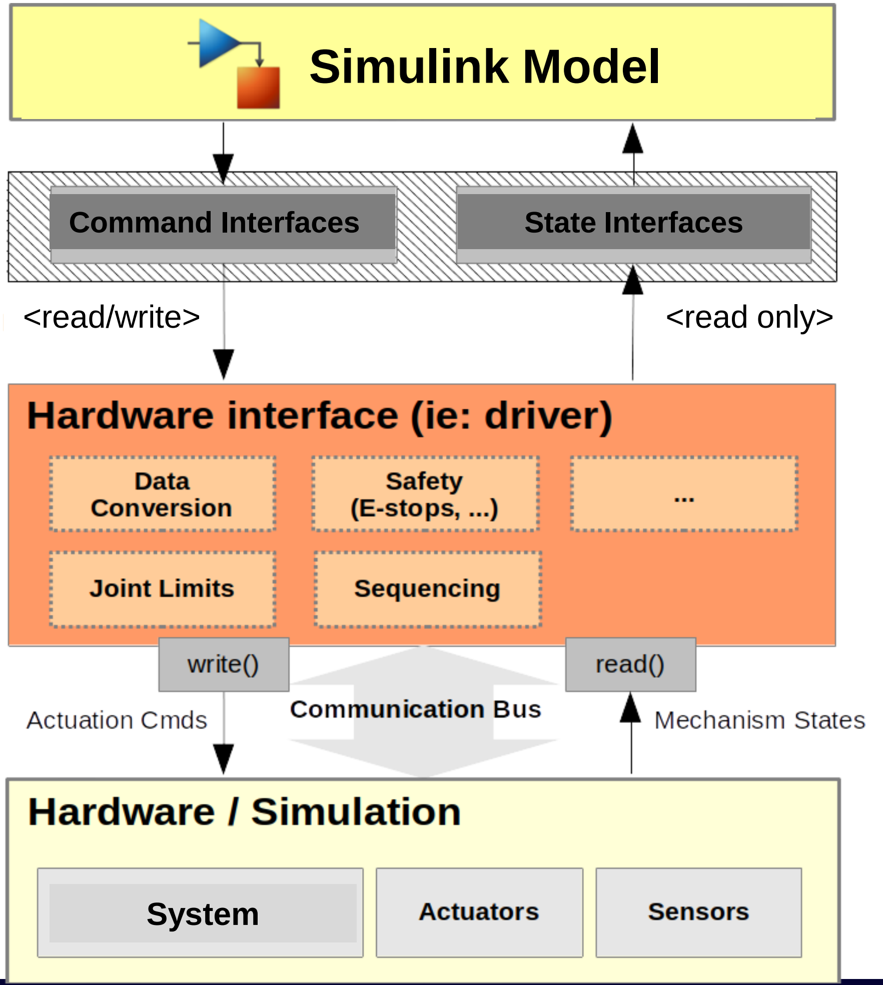




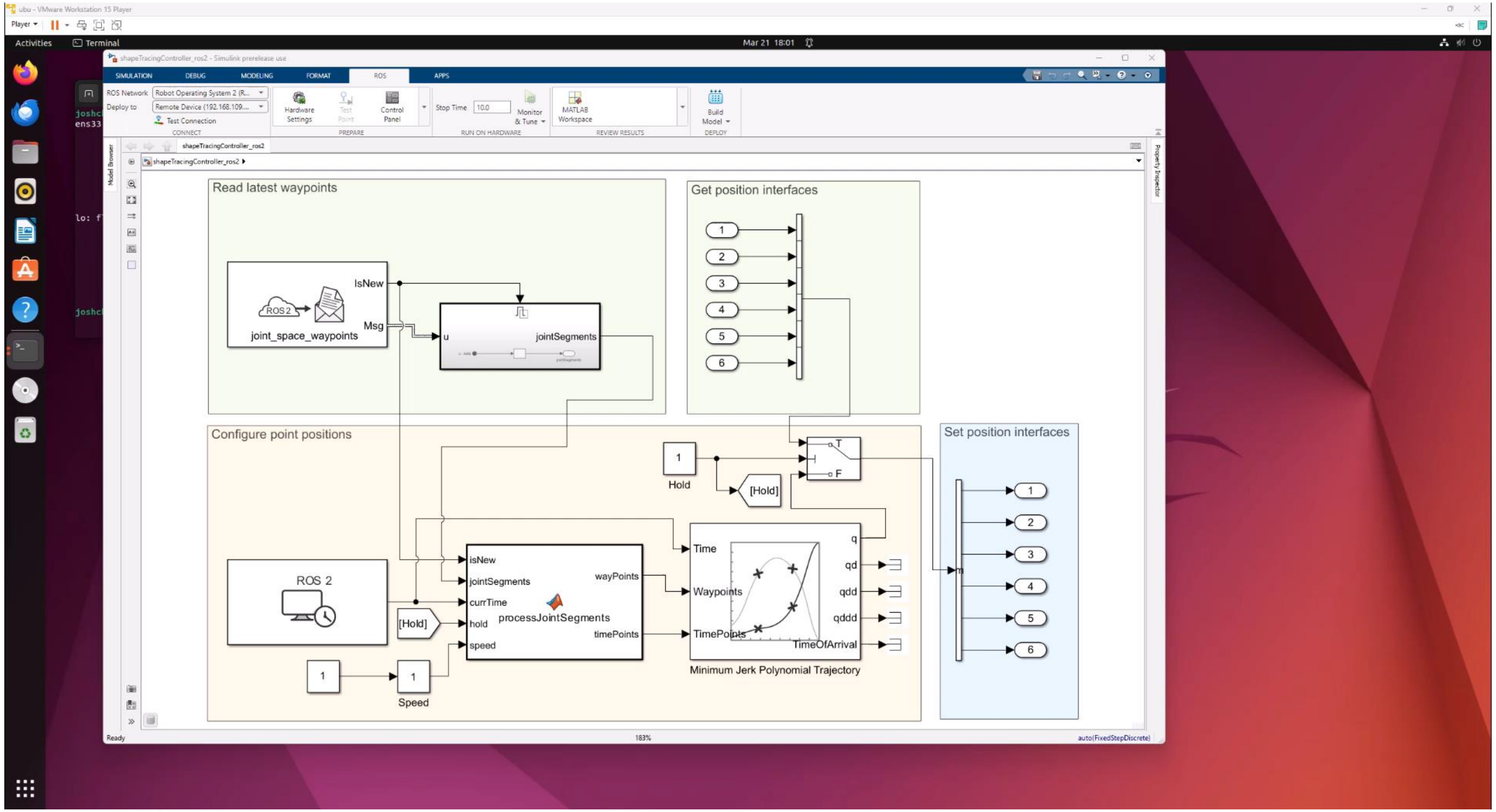
Custom implementation can be added to IRT blocks, which will generate code and get triggered at distinguished transition states.

Case Study: Model-Based Design for Painting Robots

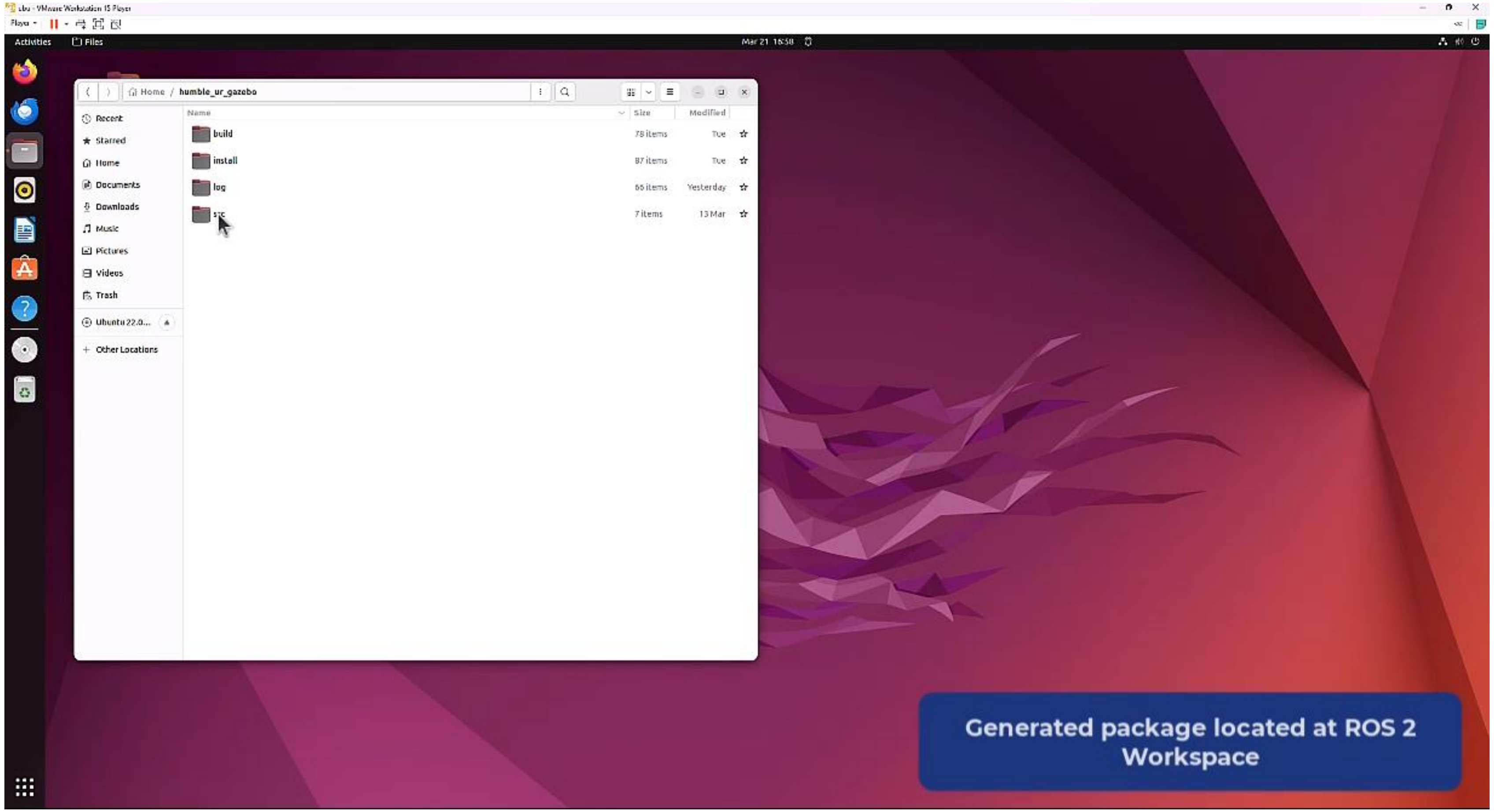




Case Study: Model-Based Design for Painting Robots



Case Study: Model-Based Design for Painting Robots

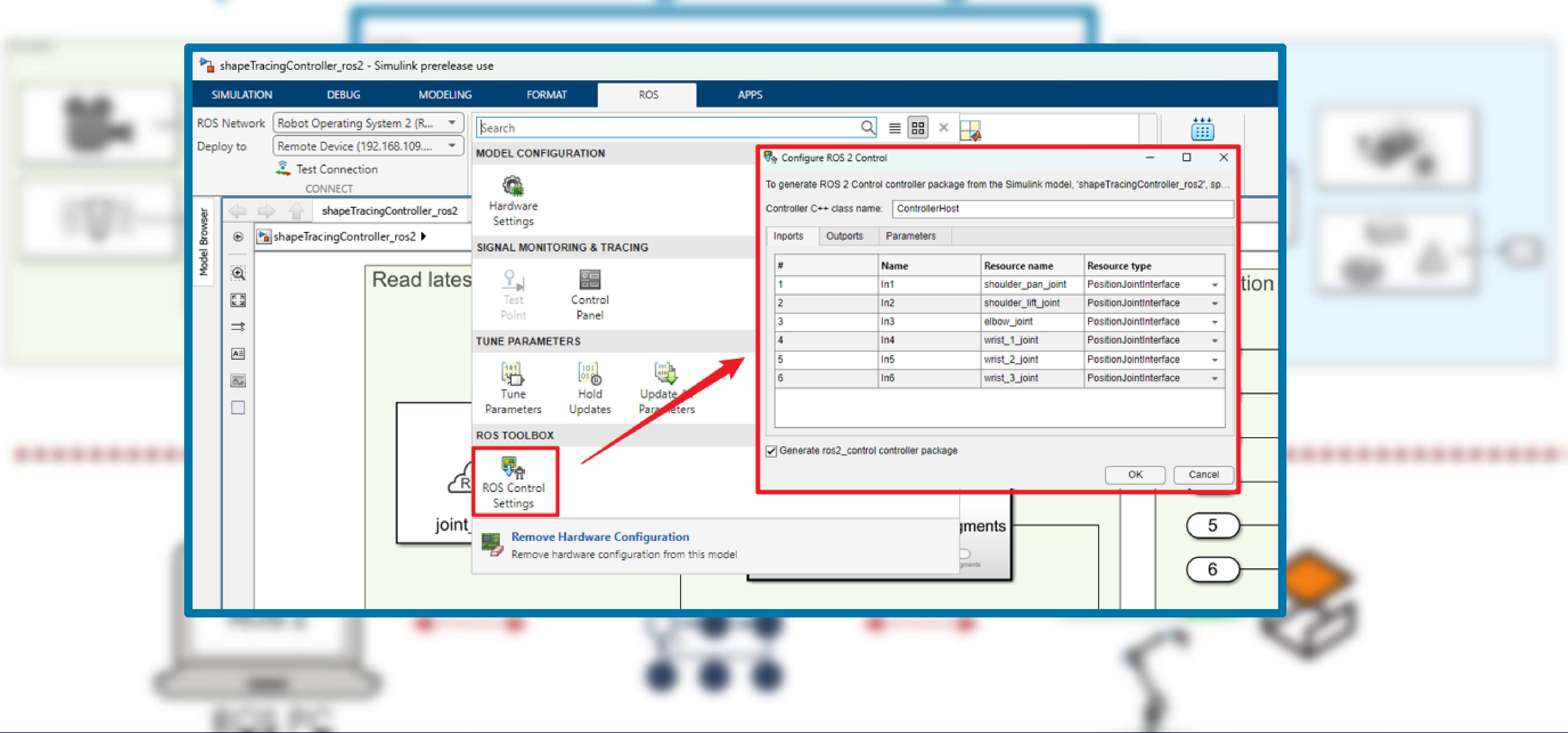


Home / humble_ur_gazebo

Name	Size	Modified
build	78 items	Tue
install	87 items	Tue
log	66 items	Yesterday
src	7 items	13 Mar

Generated package located at ROS 2 Workspace

Case Study: Model-Based Design for Painting Robots



Configure ROS 2 Control

To generate ROS 2 Control controller package from the Simulink model, 'shapeTracingController_ros2', sp...

Controller C++ class name: ControllerHost

Inports | Outports | Parameters

#	Name	Resource name	Resource type
1	In1	shoulder_pan_joint	PositionJointInterface
2	In2	shoulder_lift_joint	PositionJointInterface
3	In3	elbow_joint	PositionJointInterface
4	In4	wrist_1_joint	PositionJointInterface
5	In5	wrist_2_joint	PositionJointInterface
6	In6	wrist_3_joint	PositionJointInterface

☑ Generate ros2_control controller package

OK | Cancel

Agenda

Introduction

- Program ROS and ROS 2 using MATLAB and Simulink

From Simulink to ros2_control generation

- Automated ros2_control plugin

Summary

Key Takeaways

MATLAB and Simulink simplify ROS and ROS 2 programming

- ▶ Leverage Simulink's Model-Based Design with ros_control and ros2_control frameworks for robust controller development
- ▶ Go directly from algorithm prototyping to implementation
- ▶ Easily incorporate Simulink controllers into ros2_control framework

- **Call-To-Action:**
  - ▶ Try out the reference examples from ROS Toolbox
  - ▶ Reach out to us to work on real-world industrial applications

Learn More

MathWorks Robotics Solution Page

MATLAB and Simulink for Robotics and Autonomous Systems

Develop autonomous applications from perception to motion and optimize system-level behavior

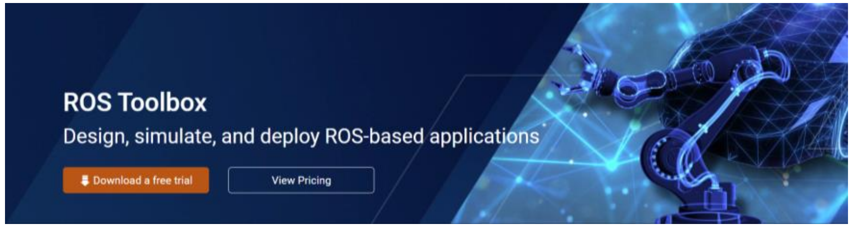
Download a free trial

Robotics Solutions

AI for Robotics

Apply AI to enable autonomy in robotics applications

AI for Robotics



ROS Toolbox

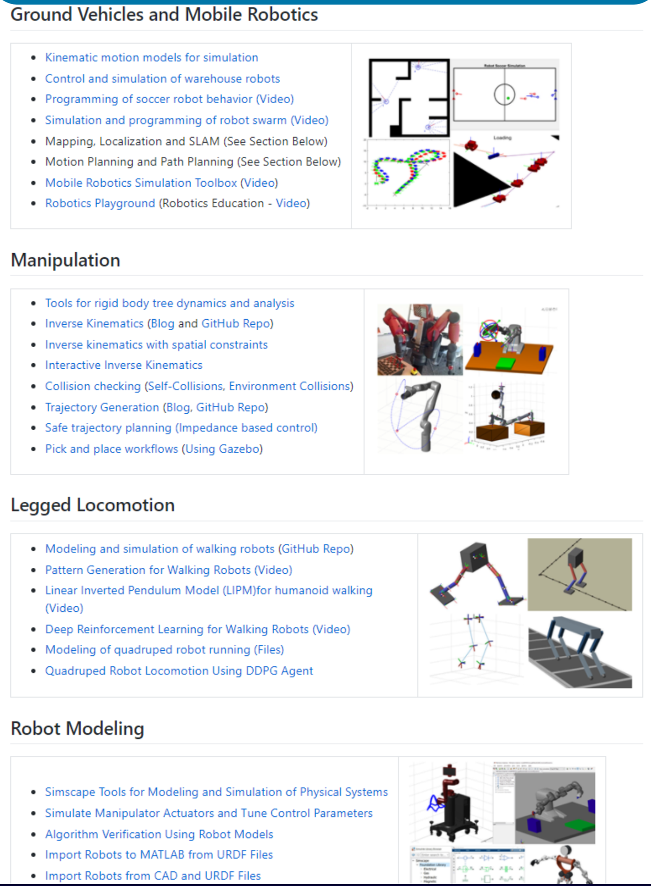
Design, simulate, and deploy ROS-based applications

Download a free trial

View Pricing

ROS Toolbox

Awesome-MATLAB-Robotics GitHub Repo (LINK)



Ground Vehicles and Mobile Robotics

- Kinematic motion models for simulation
- Control and simulation of warehouse robots
- Programming of soccer robot behavior (Video)
- Simulation and programming of robot swarm (Video)
- Mapping, Localization and SLAM (See Section Below)
- Motion Planning and Path Planning (See Section Below)
- Mobile Robotics Simulation Toolbox (Video)
- Robotics Playground (Robotics Education - Video)

Manipulation

- Tools for rigid body tree dynamics and analysis
- Inverse Kinematics (Blog and GitHub Repo)
- Inverse kinematics with spatial constraints
- Interactive Inverse Kinematics
- Collision checking (Self-Collisions, Environment Collisions)
- Trajectory Generation (Blog, GitHub Repo)
- Safe trajectory planning (Impedance based control)
- Pick and place workflows (Using Gazebo)

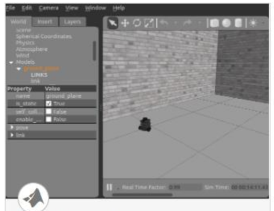
Legged Locomotion

- Modeling and simulation of walking robots (GitHub Repo)
- Pattern Generation for Walking Robots (Video)
- Linear Inverted Pendulum Model (LIPM)for humanoid walking (Video)
- Deep Reinforcement Learning for Walking Robots (Video)
- Modeling of quadruped robot running (Files)
- Quadruped Robot Locomotion Using DDPG Agent

Robot Modeling

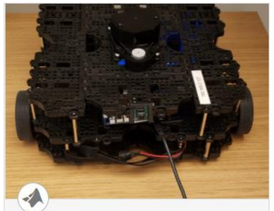
- Simscape Tools for Modeling and Simulation of Physical Systems
- Simulate Manipulator Actuators and Tune Control Parameters
- Algorithm Verification Using Robot Models
- Import Robots to MATLAB from URDF Files
- Import Robots from CAD and URDF Files

ROS Examples (LINK)



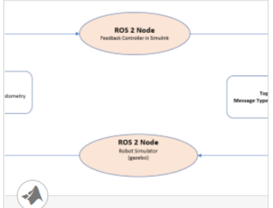
Test Robot Autonomy in Simulation

Explores MATLAB® control of the Gazebo® Simulator.



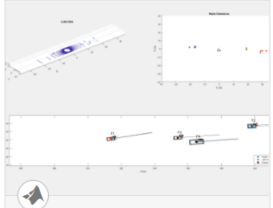
Get Started with a Real TurtleBot

Connect to a TurtleBot® using the MATLAB® ROS interface. You can use this interface to connect to a wide range of ROS-supported



Feedback Control of a ROS-Enabled Robot Over ROS 2

Use Simulink® to control a simulated robot running in a Gazebo® robot simulator over ROS 2 network.



Fusion of Radar and Lidar Data Using ROS

Perform track-level sensor fusion on recorded lidar sensor data for a driving scenario recorded on a rosbag. This example uses the same

Thank you!

YJ Lim
yjlim@mathworks.com

Josh Chen
joshchen@mathworks.com

MathWorks®

Accelerating the pace of engineering and science