



MathWorks ✓

@MathWorks

Share the EXPO experience
#MATLABEXPO



[/in/melda-ulusoy/](https://www.linkedin.com/in/melda-ulusoy/)



[/in/jennifer-gago/](https://www.linkedin.com/in/jennifer-gago/)

MATLAB EXPO

Teaching Robotics & Controls Made Easier

Melda Ulusoy, MathWorks



(She/They)

Jennifer Gago, MathWorks



(She/They)



Design Robotic Systems with MATLAB & Simulink

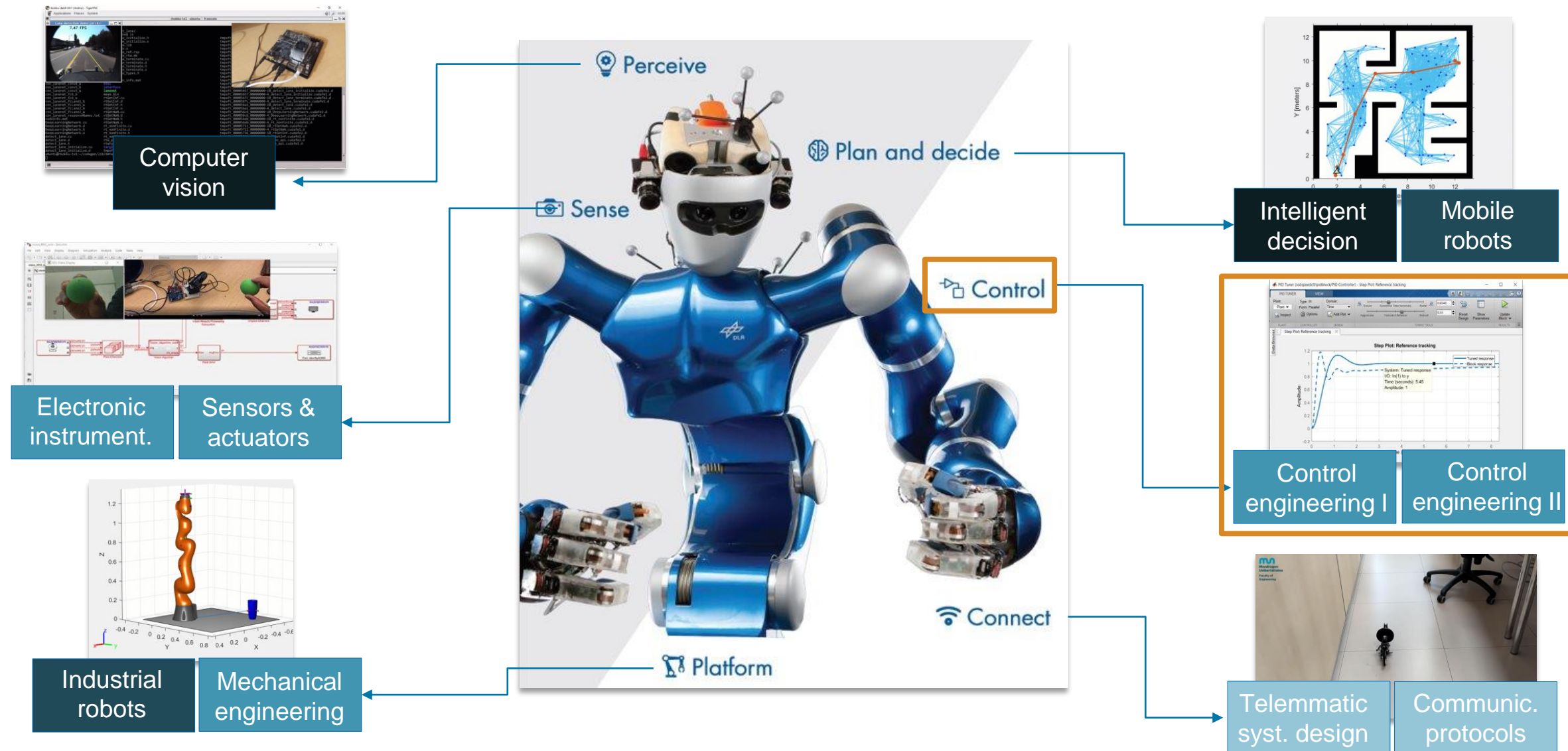
German Aerospace Centre use Model-Based Design with MATLAB & Simulink



*“**Model-Based Design** with MATLAB and Simulink covers a wide range of software domains needed for the **design of advanced robotic systems**. It enables the **simulation** of complex mechatronic systems and **controllers**, **code generation** for real-time HIL testing, **signal and image processing**, and **data analysis and visualization**.”*

Berthold Bäuml
Director, Autonomous Learning
Robot Lab,
German Aerospace Centre

Teach Robotic Systems Design with MATLAB & Simulink



Challenges that instructors face today

in teaching robotics and controls courses

How to..

- 1 Develop a curriculum under **time constraints**?
- 2 Gauge **student interest** during class?
- 3 Facilitate **independent learning**?
- 4 Scale access to resources for **hands-on learning**?
- 5 **Grade student assignments** at scale?
- 6 **Challenge students** with innovative engineering topics?



Address the teaching challenges

Today's agenda

Preparation



1

Lessons



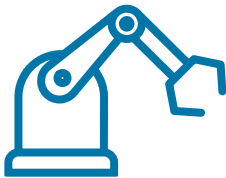
2

Study



3

Laboratories



4

Evaluation



5

Student Programs



6

Leverage ready-to-use courseware

Save time during course preparation

Preparation



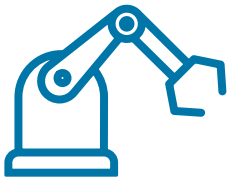
Lessons



Study



Laboratories



Evaluation



Student Programs



Explore interactive teaching content

Controls and robotics courseware

Visit our [courseware](#) for downloadable course materials including demos, tutorials and project-based learning exercises:

- [Awesome Robotics with MATLAB & Simulink](#)
- [Control Tutorials with MATLAB & Simulink](#)
- [Robotics Playground](#)
- [Applied Autonomous Robots I](#)
- [Applied Autonomous Robots II](#)
- [Mobile Robots Control](#)
- [Reinforcement Learning with MATLAB](#)
- [Electromechanical Engineering Systems](#)
- [MATLAB and Simulink ROS Tutorials](#)

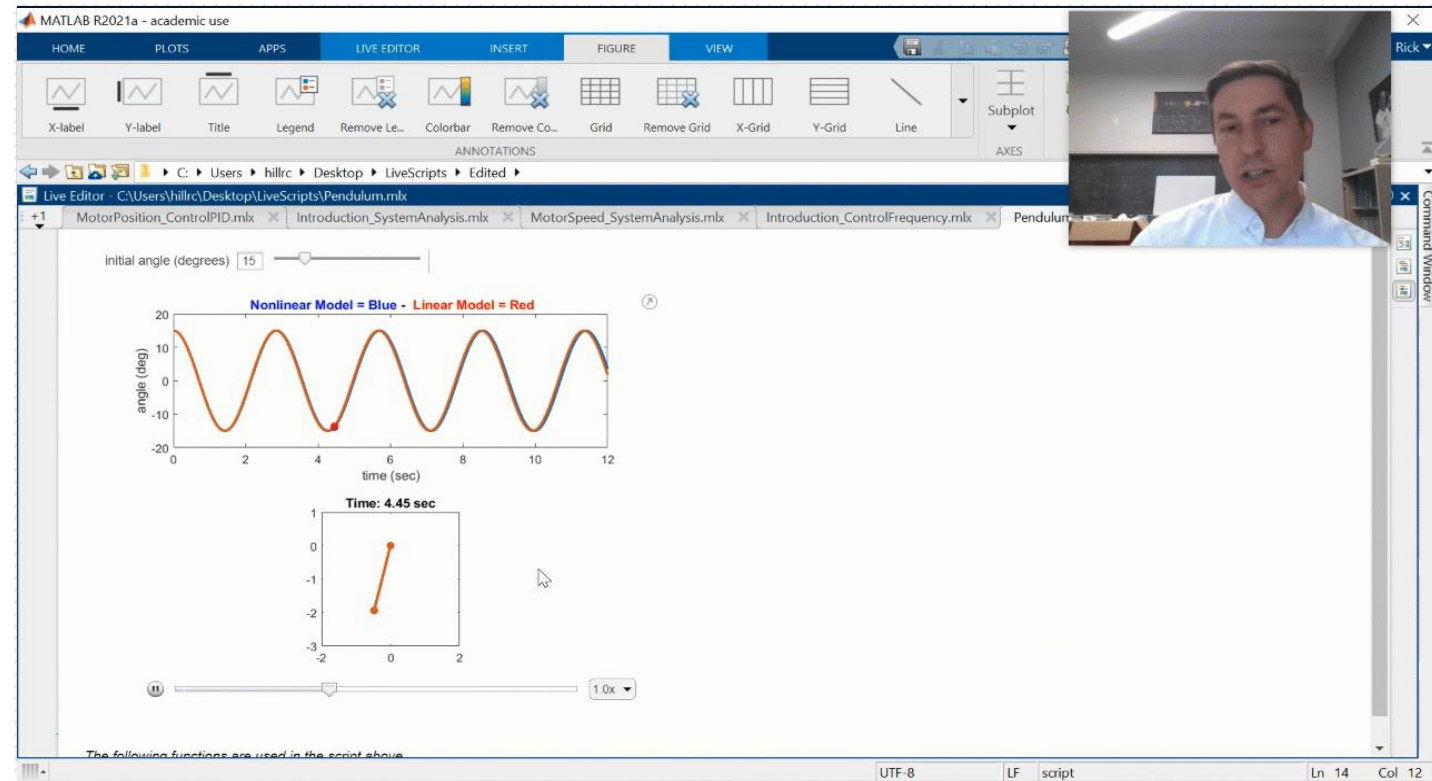
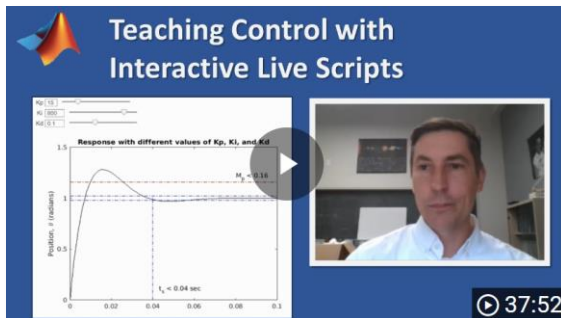
The screenshot shows the 'Teach with MATLAB and Simulink' website. The header includes a search bar and navigation links for Overview, Teach, Learn, Research, and Student Programs. The main content area features a large image of a student working on a laptop, with the text 'MATLAB and Simulink Courseware' and a description: 'Explore interactive teaching content and examples developed by MathWorks and educators from leading universities.' Below this is a dropdown menu for 'Explore courseware disciplines'. The main content is organized into a grid of six categories:

- Interactive Examples:** Fourier Analysis (Learn Fourier analysis using live scripts and MATLAB apps.)
- Labs:** Kalman Filter Virtual Lab (Try interactive exercises to study linear and extended Kalman filter design for state estimation of a simple pendulum system.)
- Assessments:** MATLAB Grader Assessment Content (Easily start adding online assessments to your courses.)
- Videos:** Tech Talks (Explore fundamental concepts in science, mathematics, and engineering.)
- Apps:** Phase Plane and Slope Field Apps (Use the Phase Plane and Slope Field apps to qualitatively analyze ordinary differential equations.)
- Books:** Thinking Like an Engineer: An Active Learning Approach, 5th Edition (This interactive courseware is designed to facilitate active learning for first-year engineering students.)

Reuse courseware developed by universities

Control tutorials by University of Michigan, Carnegie Mellon University and University of Detroit Mercy

- Using [Control tutorials](#) students can practice system modeling and analysis, control design and tuning using various examples
- Run [interactive control tutorials](#) in your browser
- [Check out this webinar](#) by Prof. Richard Hill for an overview of teaching modeling and controls with live script control tutorials



Engage your students in practical and dynamic lessons

Use interactive tools

Preparation



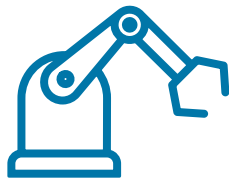
Lessons



Study



Laboratories



Evaluation

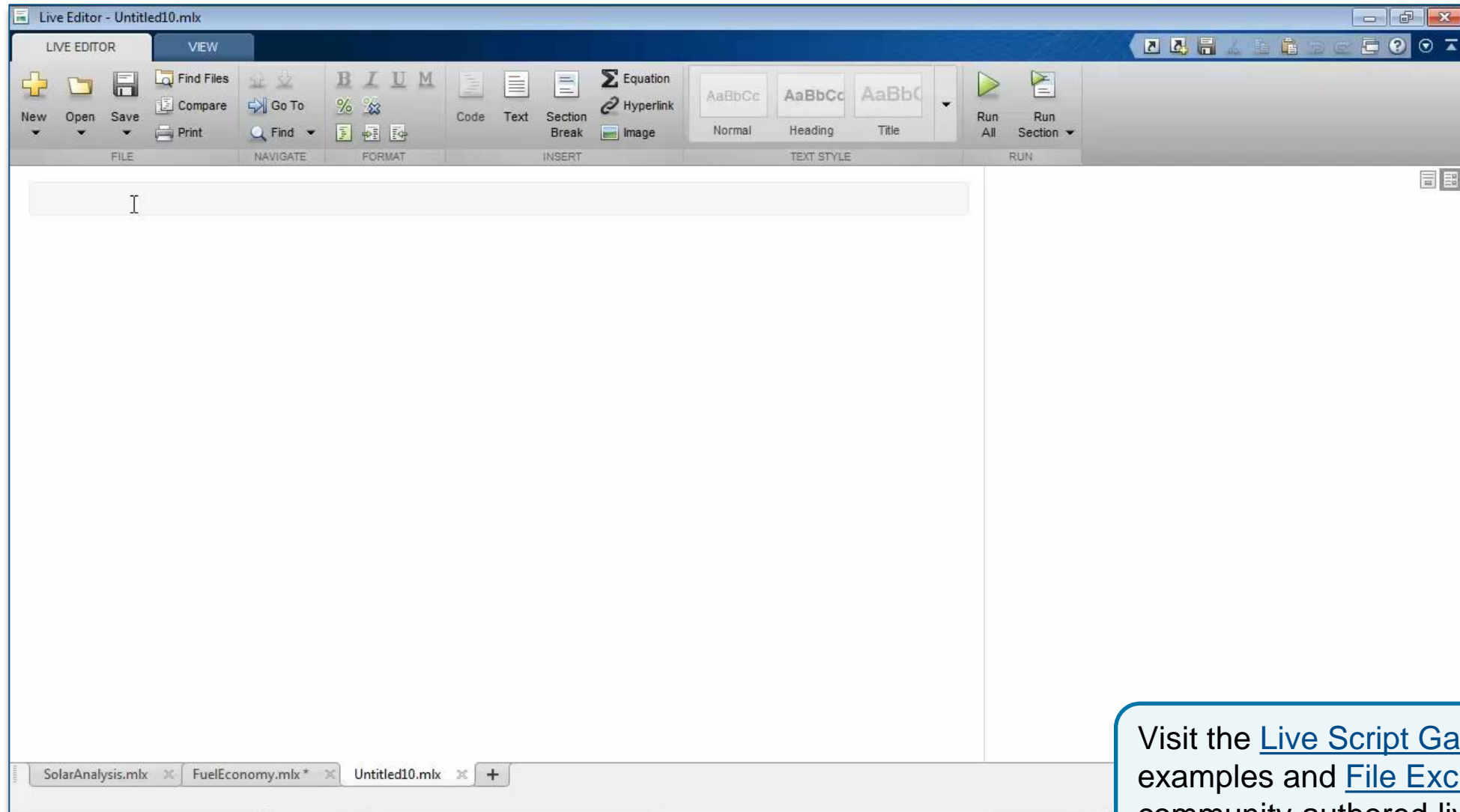


Student programs



Teach with Interactive Scripts

Live Scripts combine text, images, equations, links, code and results



Visit the [Live Script Gallery](#) for examples and [File Exchange](#) for community-authored live scripts

Get started with ROS and ROS2

Open and run Live Script documentation and tutorials

Get Started with ROS

This example introduces how to set up ROS within MATLAB, and get information about ROS network and ROS messages.

Robot Operating System (ROS) is a communication interface that enables different parts of a robot system to discover each other, and send and receive data between them. MATLAB® supports ROS with a library of functions that enables you to exchange data with ROS-enabled physical robots or robot simulators such as Gazebo®.

ROS Terminology

- A ROS network comprises different parts of a robot system (such as a planner or a camera interface) that communicate over ROS. The network can be distributed over several machines.
- A ROS master coordinates the different parts of a ROS network. It is identified by a Master URI (Uniform Resource Identifier) that specifies the hostname or IP address of the machine where the master is running.
- A ROS node contains a collection of related code and services. A ROS network can have many nodes.
- Publishers, subscribers, and services are all ROS nodes that exchange data using messages.
- A publisher sends messages to a specific topic.
- A subscriber receives those messages. A single topic can have many subscribers.

For more information, see Robot Operating System (ROS) website.

Initialize ROS Network

Use `rosinit` to initialize ROS. By default, `rosinit` creates a global node that is connected to the master. The global functions:

- 1 `rosinit`
- 2 `roslaunch`
- 3 `roscpp`
- 4 `rostopic`
- 5 `rospy`
- 6 `roswtf`
- 7 `rosls`

Use `exampleHelperROS/createSampleNetwork` to create a sample network.

Topics

Use `rostopic list` to see available topics in the ROS network. There are four active topics: `/pose`, `/roscout`, `/scan` and `/tf`. The default topics: `/roscout` and `/tf` are always present in the ROS network. The other two topics were created as part of the sample network.

Use `rostopic info <topicName>` to get specific information about a specific topic. The command below shows that `/node_1` publishes (sends messages) to the `/pose` topic, and `/node_2` subscribes (receives messages from) to that topic. See Exchange Data with ROS Publishers and Subscribers for more information.

Use `rostopic info <nodeName>` to get information about a specific node. The command below shows that `node_1` publishes to `/pose`, `/roscout` and `/tf` topics, subscribes to `/scan` topic and provides services: `node_1_get_loggers` and `node_1_set_logger_level`. The default logging services: `get_loggers` and `set_logger_level` are provided by all the nodes created in ROS network.

```

rostopic info /pose
Type: geometry_msgs/Twist
Publishers:
 * /node_1 (http://ah-av151jvtr:3201)
Subscribers:
 * /node_2 (http://ah-av151jvtr:3202)

rostopic info /scan
Type: sensor_msgs/LaserScan
Publishers:
 * /node_3 (http://ah-av151jvtr:3203)
Subscribers:
 * /node_1 (http://ah-av151jvtr:3201)
    
```

ROS

Get Started with ROS 2

This example shows how to set up ROS 2 within MATLAB, and get information about ROS 2 network and ROS 2 messages.

Robot Operating System 2 (ROS 2) is the second version of ROS, which is a communication interface that enables different parts of a robot system to discover, send, and receive data. MATLAB® support for ROS 2 is a library of functions that allows you to exchange data with ROS 2 enabled physical robots or robot simulators such as Gazebo®. ROS 2 is built on Data Distribution Service (DDS) which is an end-to-end middleware that provides features such as discovery, serialization and transportation. These features align with the design principles of ROS 2 such as distributed discovery and control over different "Quality of Service" options for transportation. DDS uses Real Time Publish-Subscribe (RTSPS) protocol which provides communication over unreliable network protocols such as UDP. For more information, see RTSPS.

To learn about ROS, see Get Started with ROS.

ROS 2 Terminology

- A ROS 2 network comprises different parts of a robot system (such as a planner or a camera interface) that communicate over ROS 2 network.
- A ROS 2 node is an entity that contains a collection of code and services. A ROS 2 network can have many nodes.
- Publishers and subscribers are different kinds of ROS 2 nodes that exchange data using messages.
- A publisher sends messages to a specific topic.
- A subscriber receives those messages. A single topic can have many subscribers.

For more information, see Robot Operating System 2 (ROS 2) website.

Initialize ROS 2 Network

Unlike ROS, ROS 2 does not require initialization in MATLAB. The ROS 2 network automatically starts with creation of nodes.

Use `ros2node` to create a node.

```

ros2node create test1
    
```

Use `ros2 node list` to see all nodes in the ROS 2 network.

```

ros2 node list
    
```

Use `clear` to shutdown the node in ROS 2 network.

```

clear test1
    
```

Use `exampleHelperROS2/createSampleNetwork` to populate the ROS 2 network with three additional nodes with sample publishers and subscribers.

```

exampleHelperROS2/createSampleNetwork
    
```

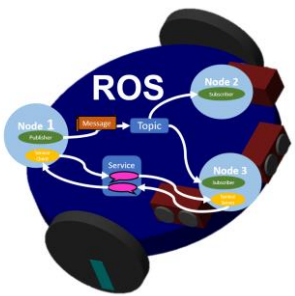
Use `ros2 node list` again, and observe that there are three new nodes, `node_1`, `node_2`, and `node_3`.

```

ros2 node list
    
```

A visual representation of the current state of the ROS 2 network is shown below. Use it as a reference when you explore this sample network in the remainder of the example.

ROS 2



MATLAB and Simulink ROS Tutorials

Teach theoretical concepts through interactive apps

This MATLAB App solves the inverse kinematics of a URDF robot model

The screenshot shows the MATLAB Inverse Kinematics Designer app. The interface is organized into several panels:

- Top Panel:** Contains a toolbar with icons for file operations (New, Open, Save, Import), scene management (Add Collision Object, Add Constraint, Edit Constraint, Report Status), solver control (Refresh Solver, Solver Settings), marker management (Marker Body, Marker Pose Constraint), and output (Check Collisions, Export).
- Scene Browser:** A tree view on the left showing the robot's hierarchical structure: Robot (base_link, base, base_link_inertia, shoulder_link, upper_arm_link, forearm_link, wrist_1_link, wrist_2_link, wrist_3_link, flange, tool0) and Scene (box).
- Scene Canvas:** A 3D plot showing the robot arm model and a brown box. The axes are labeled X, Y, and Z, with values ranging from -1 to 1.
- Scene Inspector:** A panel on the right showing the properties of the selected 'box' object.

Name	box	
Type	collisionBox	
Box X (m)	0.50	
Box Y (m)	0.50	
Box Z (m)	0.50	
X (m)	Y (m)	Z (m)
-0.5	0.5	0
Euler X (deg)	0	
Euler Y (deg)	0	
Euler Z (deg)	0	
- Constraints Browser:** A panel on the left showing preset and user-defined constraints.
 - Preset Constraints
 - Marker Pose Target [Pose]
 - User-defined Constraints
 - Constraint1 [Aiming]
- Configurations Panel:** A panel at the bottom showing a slider and a table of configurations.

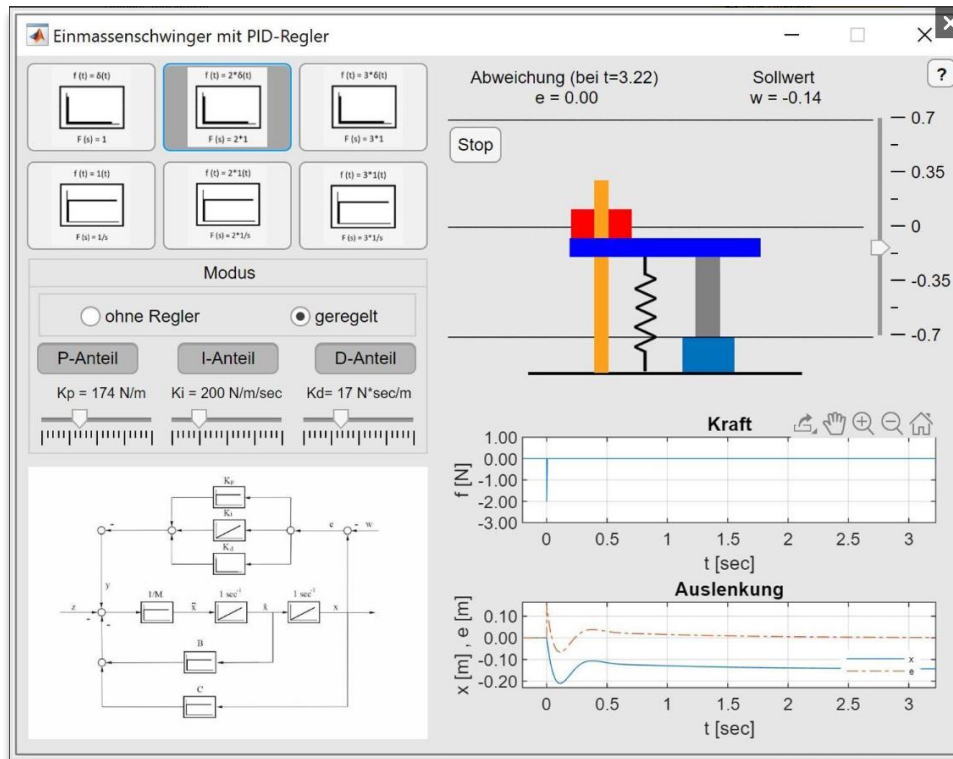
Configuration	Collision Status	Value
1 home	PASS	[0 0 0 0 0]
2 middle	PASS	[2.2567 -0.95713 0.72625 0.23088 2.2567 -4.3355e-06]
3 goal	PASS	[3.0004 0.46806 -0.7983 -2.039 4.6452 -2.4506]

Try the [Inverse Kinematics Designer](#) MATLAB App!

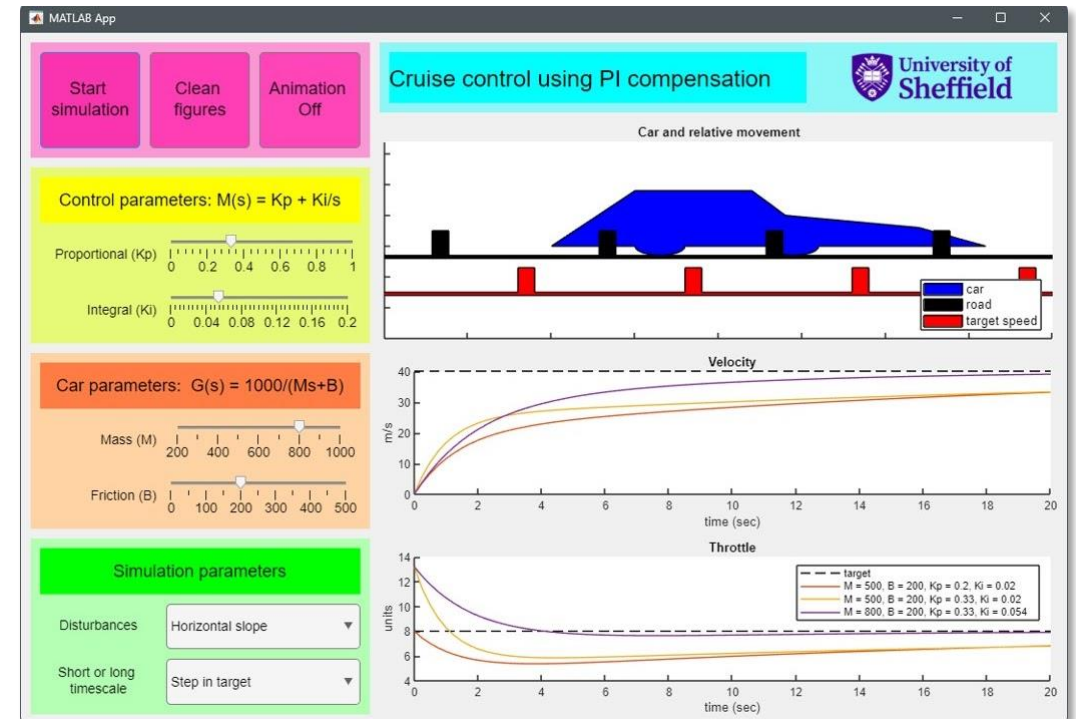
Reuse apps developed by universities

Control apps created with MATLAB App Designer

[Controls apps by RWTH Aachen University](#) allow students to study **control design and frequency response analysis** using Bode and Nyquist plots on mass-damper and inverted pendulum systems



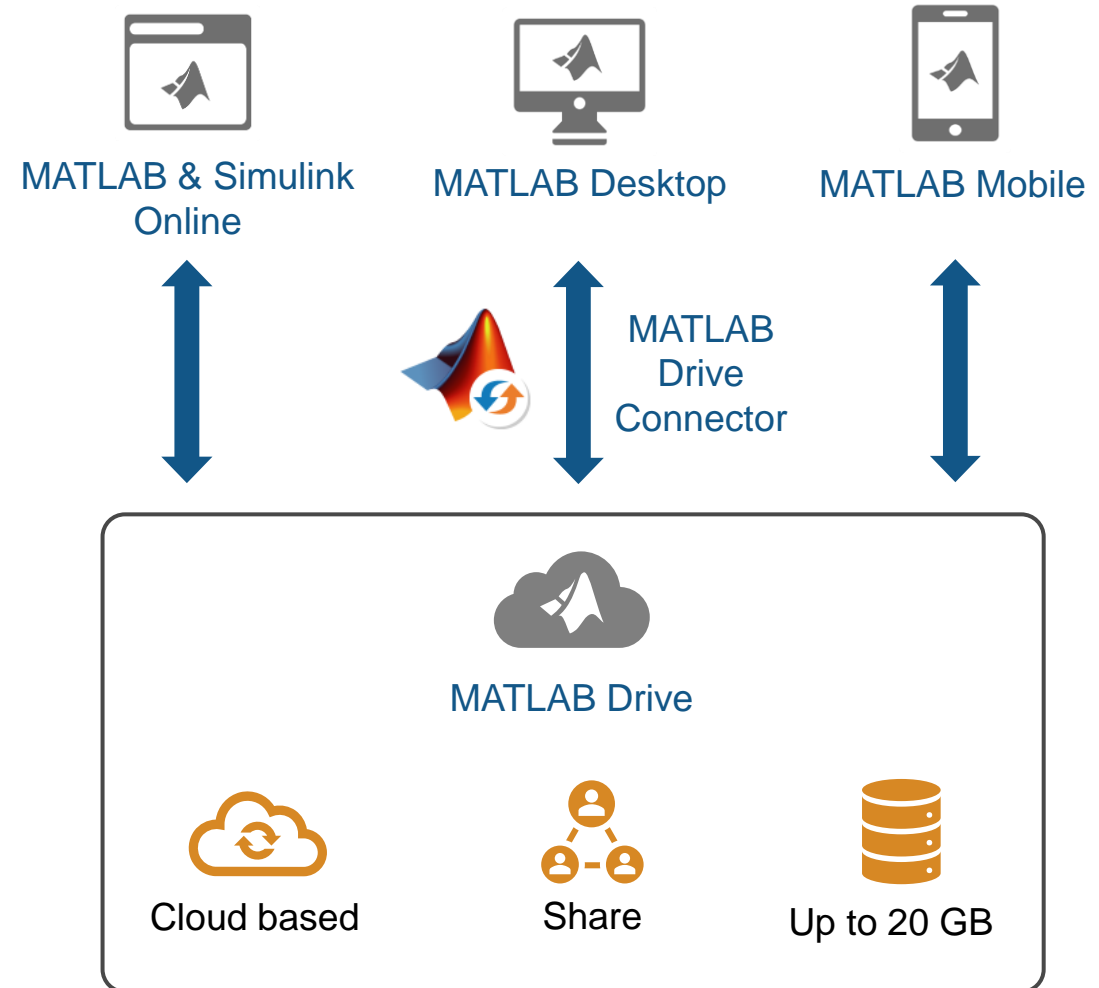
[Controls apps by University of Sheffield](#) allow students to study **dynamic system behavior and PI control design** using mixing tank and cruise control examples



Access tools and files everywhere

MATLAB & Simulink in the cloud

- Run MATLAB and Simulink in your **browser** with [MATLAB Online](#) and [Simulink Online](#)
- Run MATLAB in your **mobile devices** with [MATLAB Mobile](#)
- Collaborate and provide **cloud storage** for sharing course material using [MATLAB Drive](#) or [GitHub](#)



Explore MathWorks [cloud resources](#)

Facilitate independent learning for students

Complement in-class instruction with self-paced trainings and educational videos

Preparation



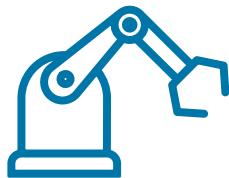
Lessons



Study



Laboratories



Evaluation



Student programs



Get started with MATLAB & Simulink for robotics and controls

Free self-paced online courses



MATLAB Onramp

14 modules | 2 hours | Languages

Get started quickly with the basics of MATLAB.



Simulink Onramp

14 modules | 2 hours | Languages

Get started quickly with the basics of Simulink.



Control Design Onramp with Simulink

7 modules | 1 hour | Languages

Get started quickly with the basics of feedback control design in Simulink.



Reinforcement Learning Onramp

5 modules | 3 hours | Languages

Master the basics of creating intelligent controllers that learn from experience.



Machine Learning Onramp

6 modules | 2 hours | Languages

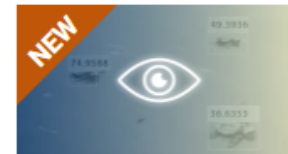
Learn the basics of practical machine learning methods for classification problems.



Deep Learning Onramp

5 modules | 2 hours | Languages

Get started quickly using deep learning methods to perform image recognition.



Computer Vision Onramp

6 modules | 2 hours | Languages

Learn the basics of computer vision to design an object detector and tracker.



Image Processing Onramp

6 modules | 2 hours | Languages

Learn the basics of practical image processing techniques in MATLAB.

Visit the [self-paced courses page!](#)

Solve practical exercises and complete projects

Free self-paced online courses

MathWorks | Training Services

Course Completion Certificate

Jennifer Gago Munoz

has successfully completed **100%** of the self-paced training course

Deep Learning with MATLAB

[Signature]
DIRECTOR, TRAINING SERVICES

03 March 2021

MathWorks | Training Services

Progress Report

Name: Jennifer Gago Munoz
Course: Deep Learning with MATLAB
Progress: 100% complete (as of 03 March 2021)

Chapters

1. Classifying Images with Convolutional Networks	100%	12. Sequence Classification Project	100%
2. Interpreting Network Behavior	100%	13. Conclusion	100%
3. Creating Networks	100%		
4. Training Networks	100%		
5. Improving Performance	100%		
6. Spectrogram Classification Project	100%		
7. Performing Regression	100%		
8. Using Deep Learning for Computer Vision	100%		
9. Classifying Sequence Data with Recurrent Networks	100%		
10. Classifying Categorical Sequences	100%		
11. Generating Sequences of Output	100%		

Release: R2020b | Language: English

Find relevant robotics and controls examples

Free self-paced online courses

The collage features several MATLAB course windows:

- Control Design Onramp with Simulink:** Shows a robotic leg model with the equation $mL^2\ddot{\theta} = \tau - mgL \sin \theta - k_d\dot{\theta}$. It includes a background section on linearization and PID control.
- Image Processing with MATLAB:** Contains sections like 'Aligning Images with Image Registration', 'Detecting Movement with the Registration Estimator', and 'Identifying the Row with the Most Strawberries'.
- Deep Learning with MATLAB:** Features 'YOLO Object Detectors' and 'YOLO Object Detection' tasks with code snippets for loading images and using the detector.
- Stateflow Onramp:** Focuses on 'Robot Vacuum Driving Modes' with an overview of sensor data and navigation tasks.
- Robot Arm Visualization:** Shows a 2D plot of a robot arm with tasks for calculating rotation angles and visualizing its path.



Project - Classify Robot Navigation

In this project, you will classify robot sensor data using a long short-term memory network.

Overview

This data set consists of two numeric sequences and one categorical vector. At every time step, there are two sensor measurements and a class label indicating which direction the robot should move. There are four classes:

- move forward
- slight left turn
- slight right turn
- sharp right turn

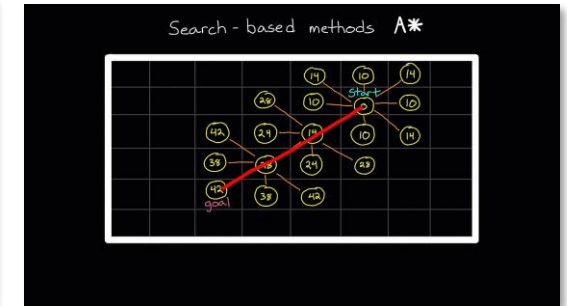
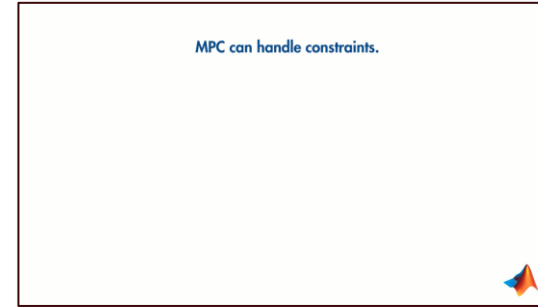
A robot is navigating a room in a clockwise direction using ultrasound sensors. The robot should follow close to the wall, without collisions.

The goal of this network is to use measurements from these sensors to determine what action the robot should take next.

Supplement your teaching of engineering concepts

MATLAB Tech Talks: Educational videos for students

- Tech talks help students gain intuition into complex engineering concepts with **easy-to-understand examples**



- Visit the [MATLAB YouTube channel](#) and [Control Systems Tech Talk Library](#) for existing and new videos!

<p>System Identification Watch videos (4 videos)</p>	<p>Fuzzy Logic Watch videos (4 videos)</p>	<p>Learning-Based Control Watch videos (3 videos)</p>	<p>Robust Control Watch videos (5 videos)</p>	<p>Control Systems in Practice Watch videos (14 videos)</p>	<p>Understanding Model Predictive Control Watch videos (9 videos)</p>	<p>Understanding PID Control Watch videos (7 videos)</p>	<p>Understanding Kalman Filters Watch videos (7 videos)</p>
<p>State Space Watch videos (4 videos)</p>	<p>Reinforcement Learning Watch videos (7 videos)</p>	<p>Trimming and Linearization Watch videos (2 videos)</p>	<p>Drone Simulation and Control Watch videos (5 videos)</p>	<p>Understanding Control Systems Watch videos (6 videos)</p>	<p>Understanding Bode Plots Watch videos (4 videos)</p>	<p>Using Bode Plots Watch videos (5 videos)</p>	<p>Autonomous Navigation Learn the fundamental concepts of navigation for autonomous systems.</p>

Scale access to resources for hands-on learning

Build virtual and low-cost hardware labs

Preparation



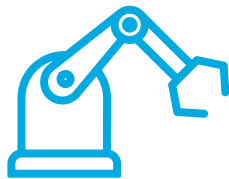
Lessons



Study



Laboratories



Evaluation

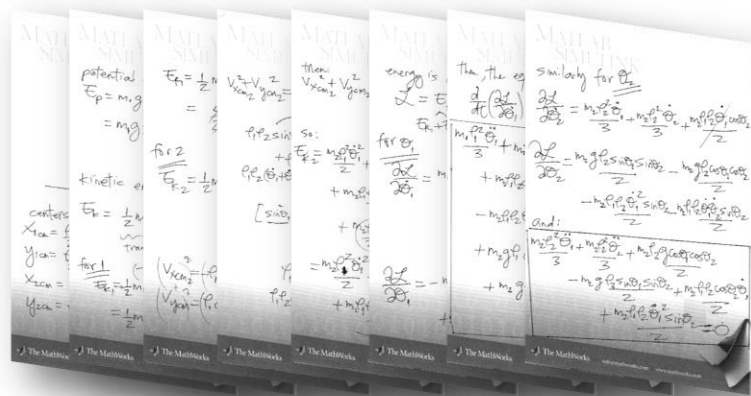


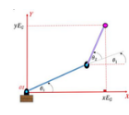
Student Programs



Build a virtual lab from written notes

MATLAB, Simulink and Simscape help you design practical applications based on core concepts





Inverse Kinematics of a 2-link Robot Arm

version 1.0.0 (2.29 MB) by Mihir Acharya **STAFF**

Calculate and visualize the inverse kinematics of a 2-link robot arm along with the Jacobian, and make the robot to write Hello.

★★★★★ (2)

2.3K Downloads

Updated 11 Jan 2019

[View License](#)

+ Follow
Download

Download this example from [File Exchange](#)!

MATLAB

Inverse kinematics of a two-link robot arm:

$$x_E(L_1, L_2, \theta_1, \theta_2) = L_2 \cos(\theta_1 + \theta_2) + L_1 \cos(\theta_1)$$

$$y_E(L_1, L_2, \theta_1, \theta_2) = L_2 \sin(\theta_1 + \theta_2) + L_1 \sin(\theta_1)$$

```
% Symbolic variables for the end effector kinematic
syms L_1 L_2 theta_1 theta_2
% End effector as function of 4 parameters
xE = L_2*cos(theta_1+theta_2) + L_1*cos(theta_1)
```

$$x_E = L_2 \cos(\theta_1 + \theta_2) + L_1 \cos(\theta_1)$$

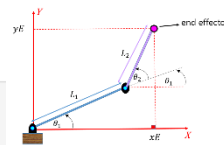
$$y_E = L_2 \sin(\theta_1 + \theta_2) + L_1 \sin(\theta_1)$$

Once you know the length of the links, the equations become easier

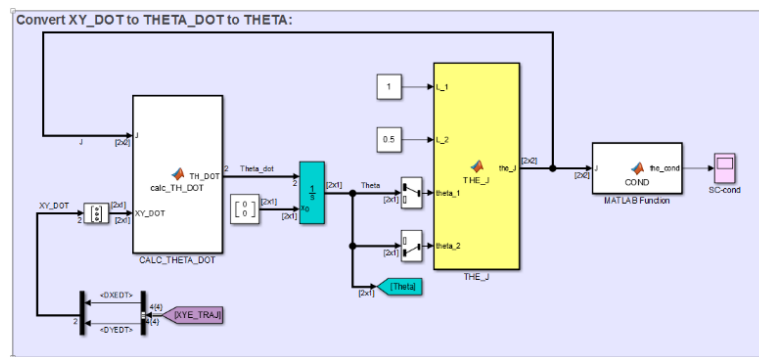
$$L_1, L_2 \Rightarrow x_E(\theta_1, \theta_2)$$

$$L_1, L_2 \Rightarrow y_E(\theta_1, \theta_2)$$

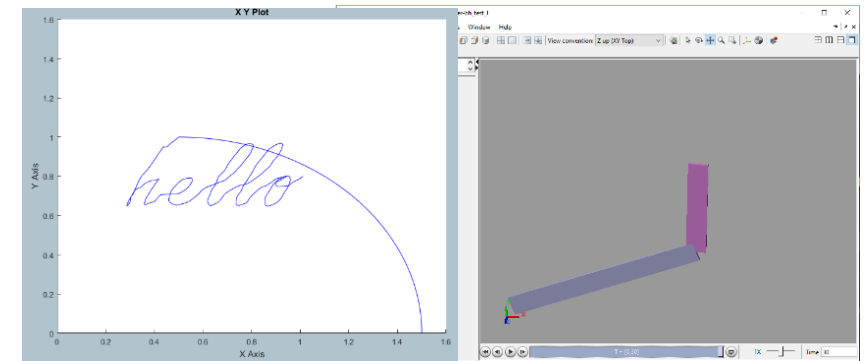
```
% End effector as function of 2 parameters
fxE(theta_1, theta_2) = subs(xE,[L_1 L_2],[4 2])
fxE(theta_1, theta_2) = 2*cos(theta_1) + 4*cos(theta_1)
fyE(theta_1, theta_2) = subs(yE,[L_1 L_2],[4 2])
fyE(theta_1, theta_2) = 2*sin(theta_1) + 4*sin(theta_1)
```



Simulink



Simscape



Simulate multiple virtual environments

Virtual labs using Simulink with additional tools

Automated driving



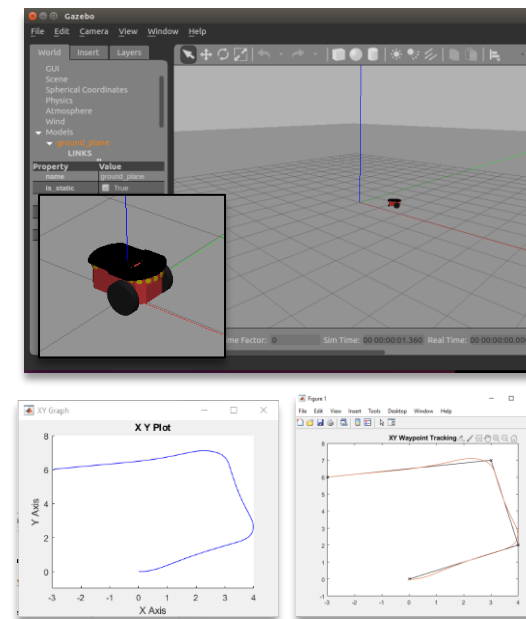
RoadRunner

VR environments



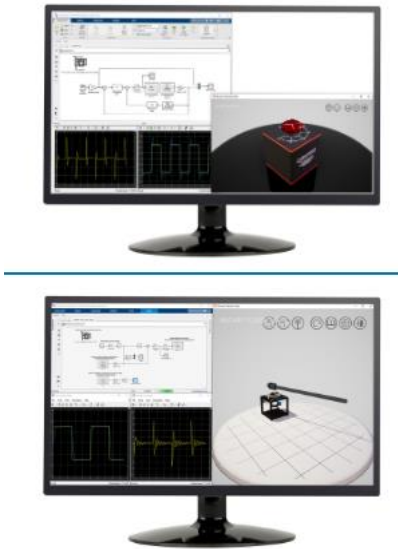
Simulink 3D Animation

Co-Simulation



Gazebo

Digital twins



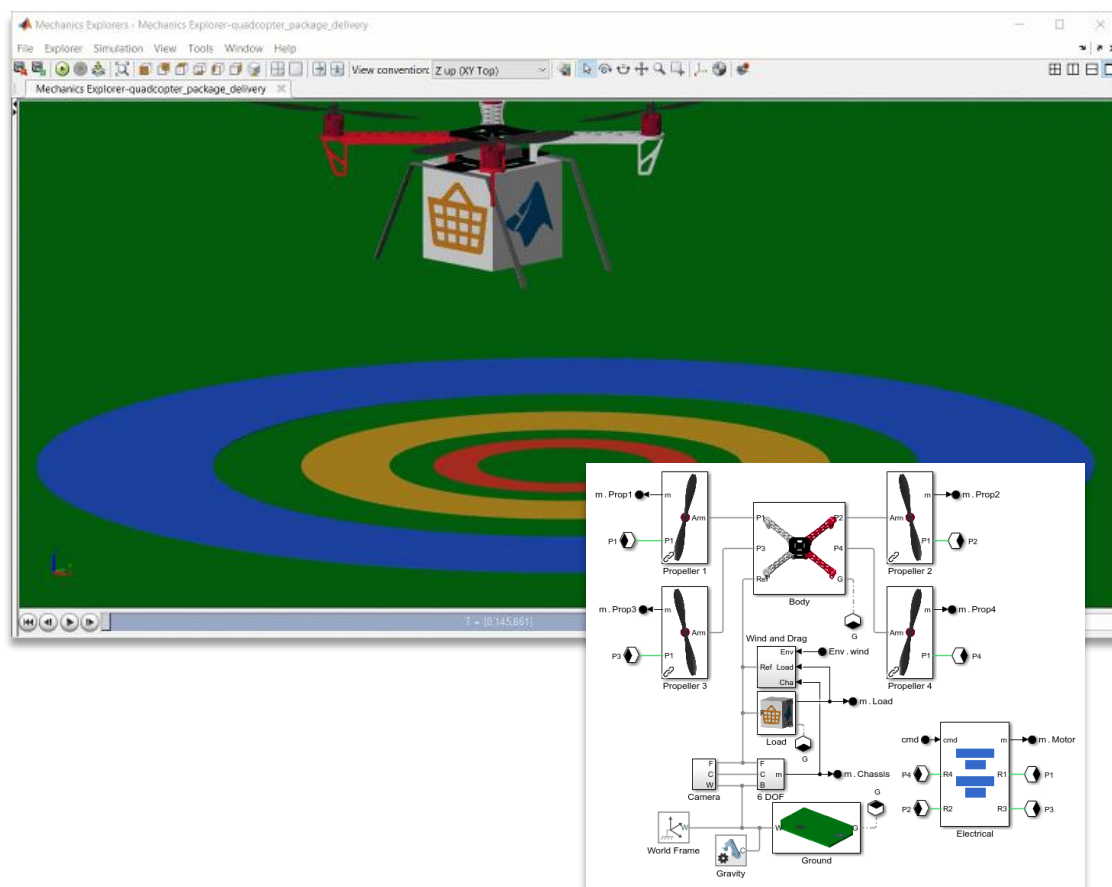
Quanser Labs



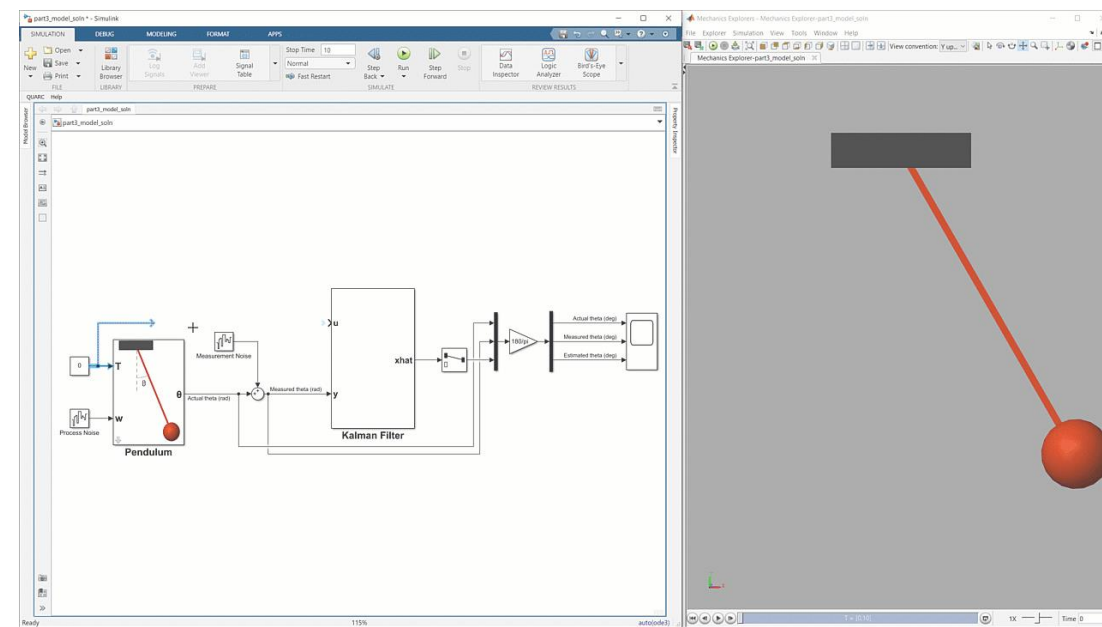
Use virtual labs based on physical models

Simscape-based labs by MathWorks

This [Delivery Quadcopter example](#) shows how to model the control, mechanical and electrical systems of a UAV.



This [Kalman Filter Virtual Lab](#) contains interactive exercises for students to study linear and extended Kalman filter design.

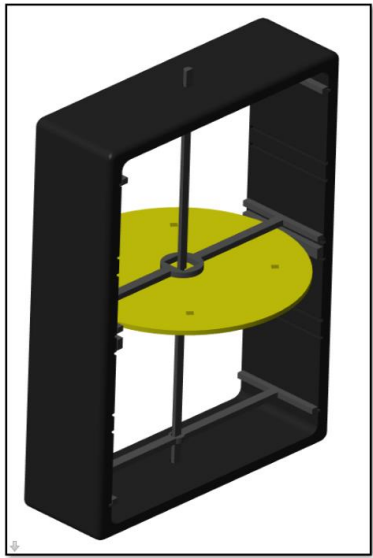


*Solutions available upon instructor request.
Contact us at onlineteaching@mathworks.com

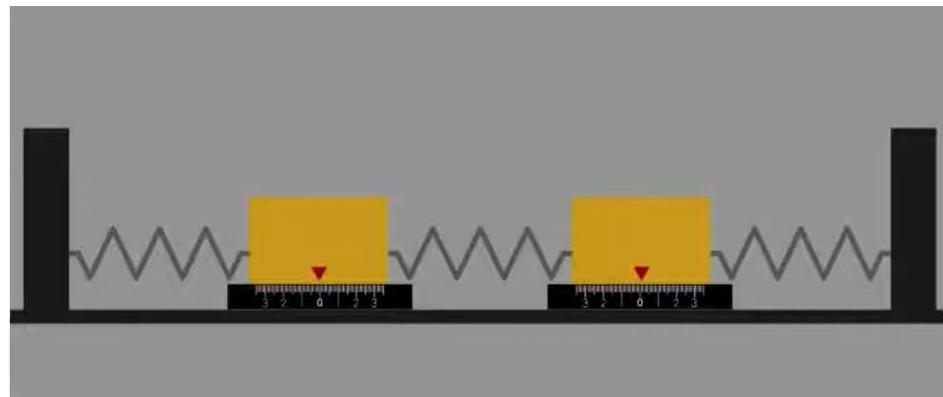
Reuse virtual labs developed by universities

Simscape-based labs by Kennesaw State University

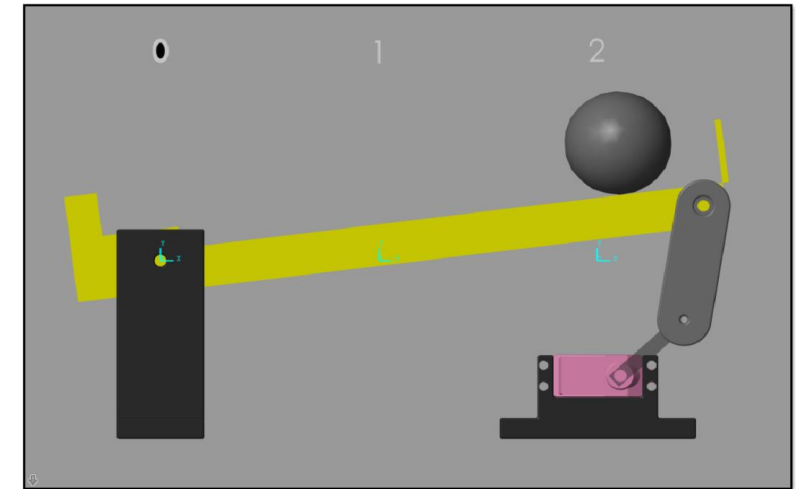
- Vibration and control virtual labs provide various virtual mechanisms that students can use to study topics such as free and forced response of multi dof systems, mode ratios and pid control
- [Download](#) vibration and control virtual labs and supporting curriculum materials



PID Controlled disc



Mass-spring system



PID Ball control

Connect MATLAB & Simulink to hardware

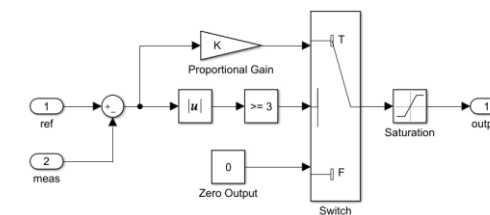
Hardware Support



```

if inp >= 1
    out = K*inp;
elseif inp < 0
    out = 0;
else
    out = 1;
end

```



MATLAB

Simulink



Automated code generation
 Coder/Embedded Coder (C/C++),
 HDL Coder, PLC Coder, GPU Coder

Check our [hardware support page!](#)

Use low-cost hardware for project-based learning

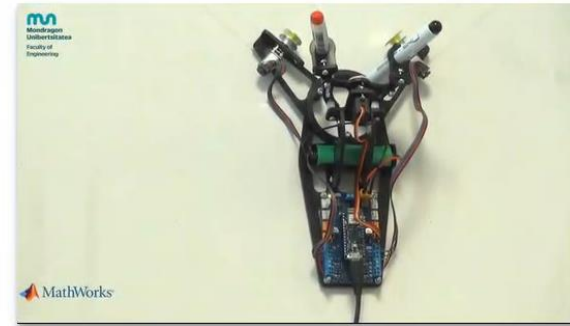
Arduino Engineering Kit curriculum by Mondragon University

- 📁 C01_Kinematics
- 📁 C02_Math_computations
- 📁 C03_Closed_lool_control
- 📁 C04_Model-based_design
- 📁 C05_Simulations
- 📁 C06_Robot_movement
- 📁 C07_Diferencial_drive
- 📁 C08_Path_following_algorithm
- 📁 C09_Image_processing
- 📁 C10_Wireless_networks
- 📁 C11_Coordinate_geometry
- 📁 C12_Trigonometry

Self-balancing motorcycle



Drawing robot



Autonomous rover



File Exchange

MATLAB Central ▾ | Files | Authors | My File Exchange ▾ | Publish | About



ARDUINO-ENGINEERING-KIT

version 1.0.1 (78.5 MB) by Gaizka Bellido Sanjulian

Download this example from
[GitHub](#) or [File Exchange](#)!

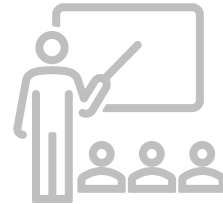
Assess learning at scale and get immediate feedback

Auto-grade assignments

Preparation



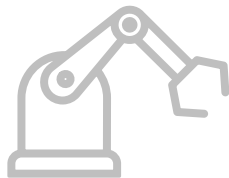
Lessons



Study



Laboratories



Evaluation



Student Programs



Auto-grade student work with MATLAB Grader

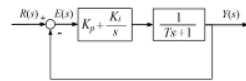
System Dynamics and Controls Problem Collection



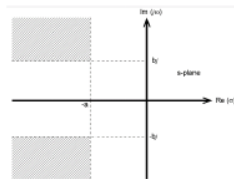
LAB SESSION 1 – CONTROL ENGINEERING I

PI CONTROLLER DESIGN WITH MATLAB

In this problem, you will write a function that designs and returns a PI controller for the system illustrated below:



It is desired that the closed-loop poles have a real part less than $-a$ and an imaginary part greater than b (or less than $-b$). Your function should accept the two parameters, a and b , as input, along with the first-order plant time constant, T . Modify the solution template by adding formulas to compute the proportional and integral control gains, K_p and K_i , from so that the poles of the closed-loop system lie in the shaded region of the complex s -plane shown below:



The function declaration is given below.

```
function [Kp,Ki,wn] = PIcontrol(a,b,T)
Kp = ;
Ki = ;
end
```

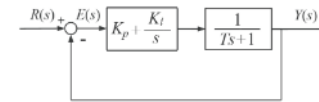
Code is also provided to test your function:

```
a = 5;
b = 7;
T = 1;
```

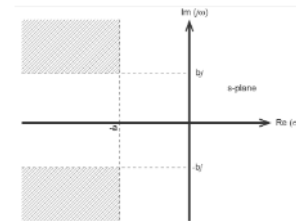


Design a PI Controller

In this problem you will write a function that designs and returns a PI controller for the system illustrated below:



It is desired that the closed-loop poles have real part less than $-a$ and imaginary part greater than b (or less than $-b$). Your function should accept the two parameters, a and b , as input, along with time constant of the first order plant, T . Modify the solution template by adding formulas to calculate the proportional and integral control gains, K_p and K_i , such that the poles of the closed-loop system are placed in the shaded region of the complex s -plane shown below:



The function declaration has been provided for you in the solution template. Code has also been provided to test your function in the 'Code to call your function' box.

Function

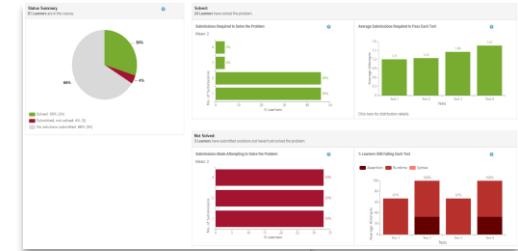
Reference Solution Save Reset MATLAB Documentation

```
1 function [Kp,Ki,wn] = PIcontrol(a,b,T)
2
3
4 Kp = ;
5
6 Ki = ;
7
8 end
```

Code to call your function

Reset

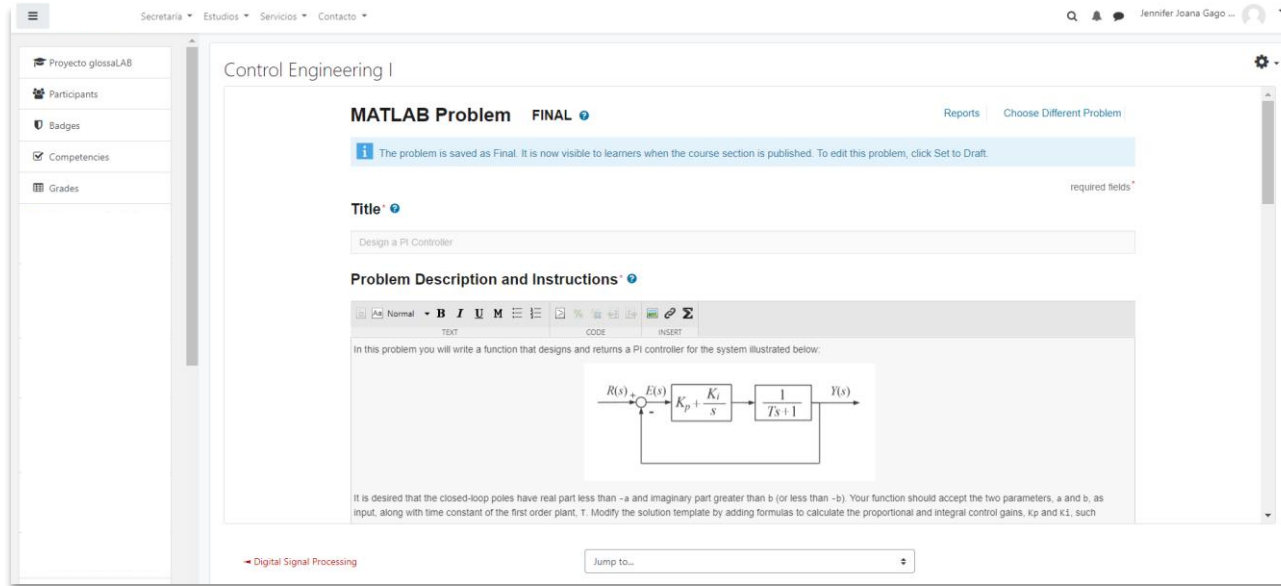
```
1 a = 5;
2 b = 7;
3 T = 1;
4 [Kp,Ki,wn] = PIcontrol(a,b,T)
```



Use [MATLAB Grader](#) and explore the [problem collection](#)!

Integrate MATLAB Grader with your LMS

Interactive homework on Moodle, Blackboard, Canvas...



Grades

Grade Item	Calculated weight	Grade	Range
Projecto glossaLAB			
Linear Algebra	-	-	0-100
Physics	-	-	0-100
Digital Signal Processing	-	-	0-100
Curso "Enseña con MATLAB" (2h)	-	-	0-100
Control Engineering I	-	-	0-100
Course total Weighted mean of grades. Include empty grades.	-	-	0-100



Create interactive course assignments



Automatically grade student work and provide feedback



Run your assignments in any learning environment

Integrate MATLAB Grader with your LMS



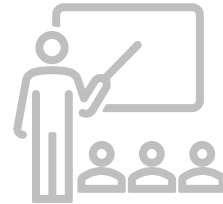
Challenge students with innovative engineering topics

Leverage student programs

Preparation



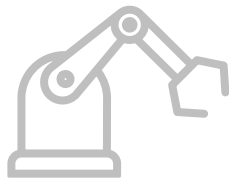
Lessons



Study



Laboratories



Evaluation



Student Programs



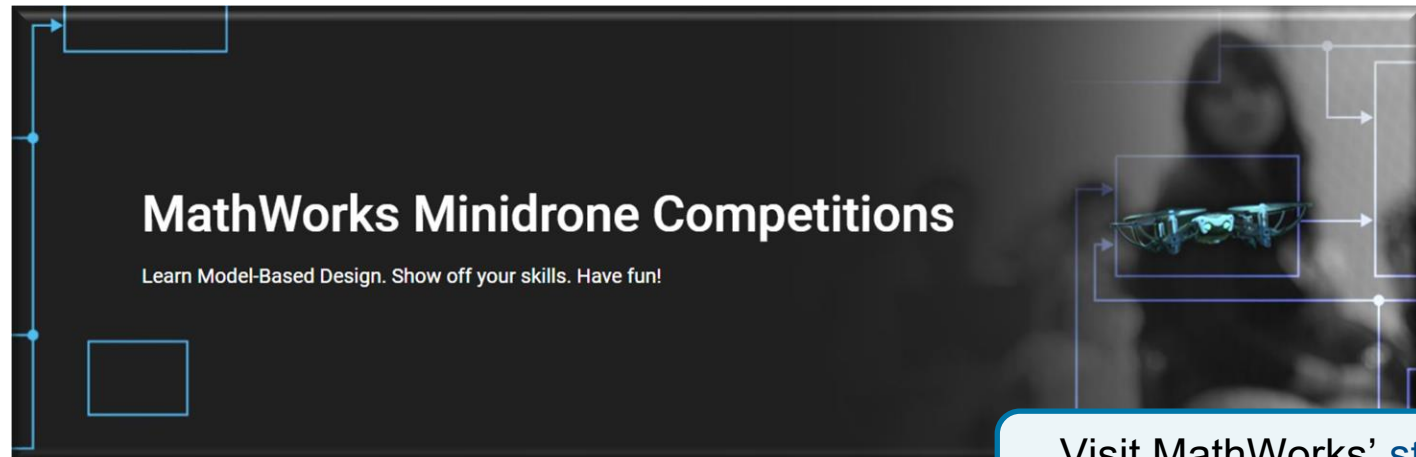
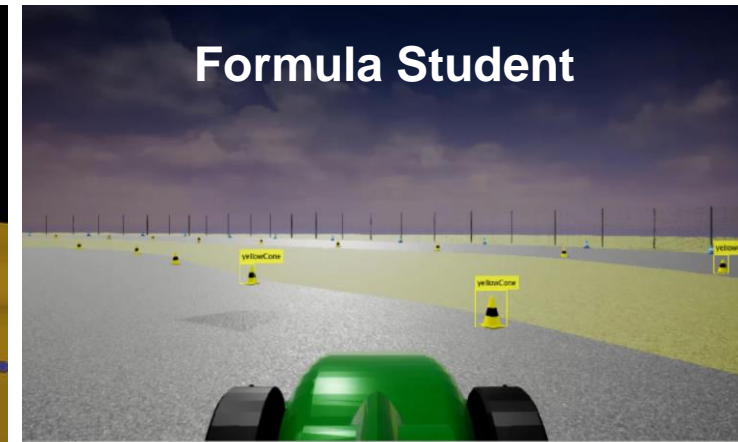
Encourage students to put their knowledge to the test

Robotics competitions supported by MathWorks



Robotics

- BEST Robotics
- Brain-Computer Interface
- Collegiate Wind Competition
- European Rover Challenge
- FIRST Robotics
- Intelligent Ground Vehicle Competition
- Korea Semiconductor Design Challenge
- Micromouse Contest
- National DD-Robocon
- Pan-African Robotics Competition
- Road2FEI
- RobAFIS
- ROBO-ONE
- RoboCup
- RoboCupJunior
- RoboNation Competitions
- RoboRace
- Singapore Autonomous Underwater Vehicle Challenge
- VEX Robotics
- World Robot Summit



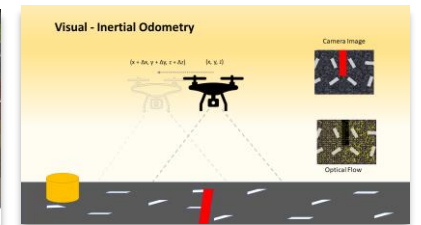
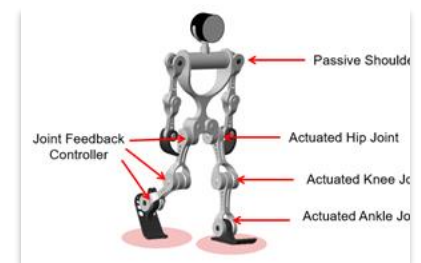
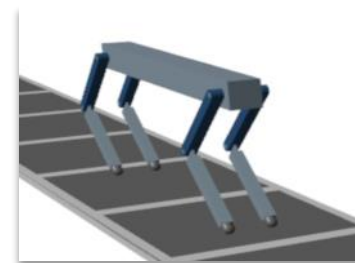
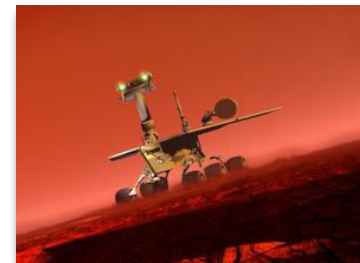
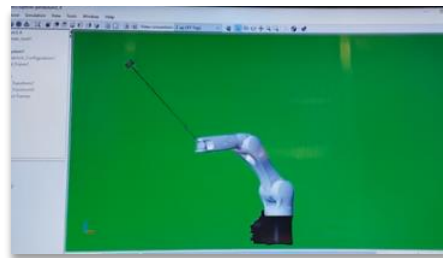
Visit MathWorks' [student competitions webpage](#)

Challenge your students

Ideas for research projects, undergraduate/postgraduate final projects...



MathWorks Excellence in Innovation Robotics Projects



[View](#) the 2022 Simulink Student Challenge winners



Overcome challenges in teaching robotics and controls

Try these MATLAB & Simulink based tools

Preparation

Courseware

Lessons

Interactive notebooks

Cloud resources

Apps

Study

Self-paced online trainings

Tech talks

Laboratories

Virtual labs

Hardware support

Evaluation

Auto-grading

Student Programs

Competitions

Innovation Projects

Build practical engineering skills with MATLAB & Simulink

Mondragon University use Project-Based Learning to teach robotics and controls



*“**Campus-wide access** to MATLAB and Simulink enabled us to develop and implement an educational model founded on **practical, project-based learning** that helps students go from merely knowing engineering concepts to knowing how to **apply them**.”*

***Companies** that use simulation already and companies looking to lower costs by introducing simulation recognize the benefits of **hiring graduates** who have practical **experience with MATLAB and Simulink**.”*

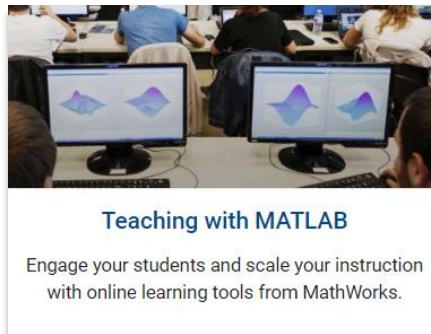
Carlos García

R&D Manager, Faculty of Engineering
Mondragon University

Get started quickly with these teaching resources

Call-to-action

1. Learn more about teaching tools and resources
Complete the [“Teaching with MATLAB” course](#)



2. Visit courseware site for teaching materials
For [controls](#) and [robotics](#)



Contact MathWorks for **curriculum development**
Email us at academicsupport@mathworks.com

MATLAB EXPO

Thank you



© 2023 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Join other **MATLAB EXPO** sessions and demos:

- Interactive Learning with MATLAB Apps, Live Scripts, and MATLAB Grader
- LMS-Based Teaching Tools for Educators