Enseigner la Robotique avec MATLAB & Simulink

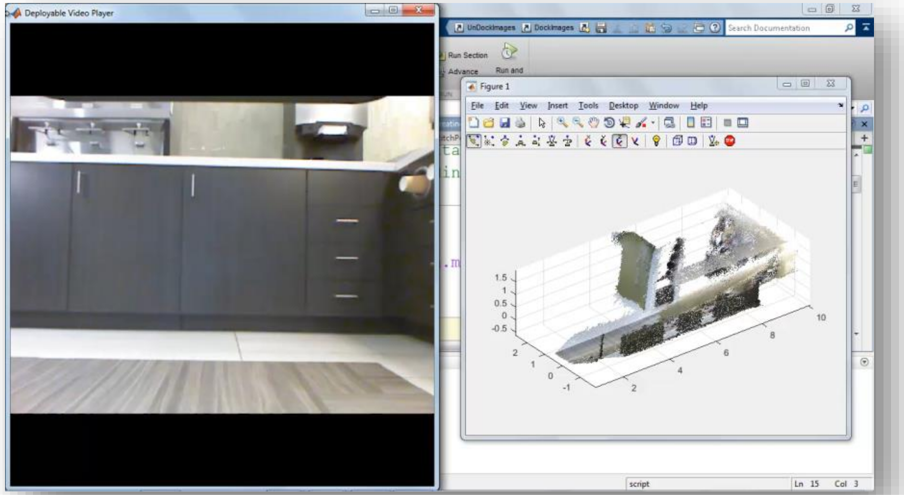
Valerie Leung
Application Engineer

Ninon Candalh-Touta
Ingénieure Pédagogique

Developing Autonomous Robotics and Navigation Systems

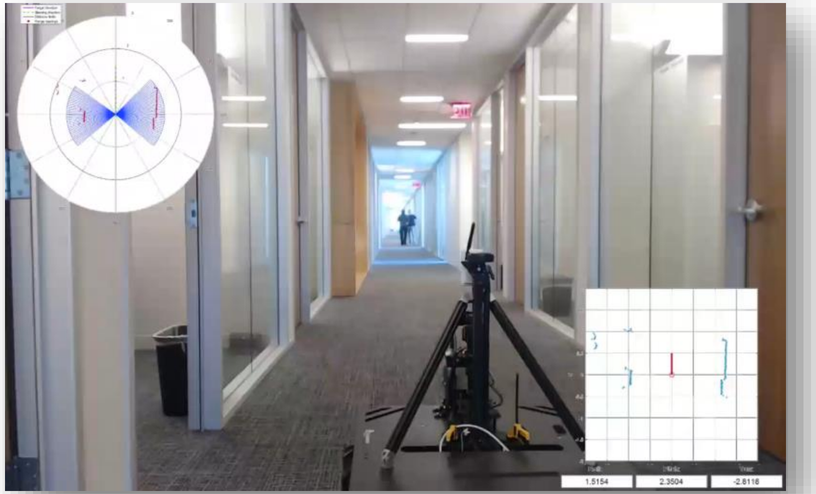
Valerie Leung
Application Engineer

Examples of Autonomous Systems

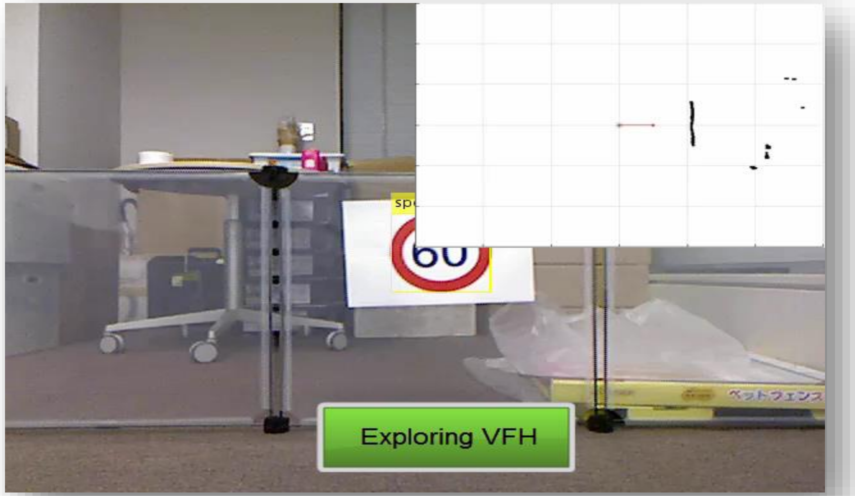


Mapping the Environment

Manipulator Arms/Humanoids



Autonomous Ground Robots



Machine Learning and Controls

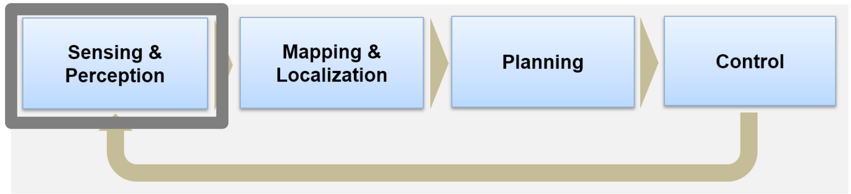
Key Components for Enabling Full Autonomy

Physical Systems & Environment

Algorithms

Sensing & Perception → Mapping & Localization → Planning → Control

Examples of automated driving sensors



Camera

Radar-based object detector

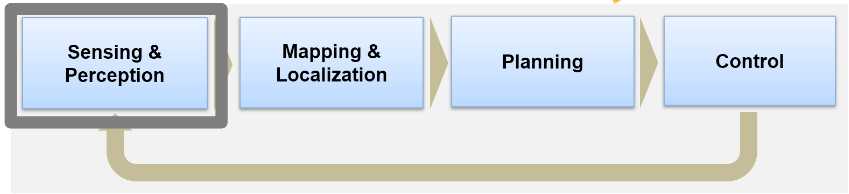
Vision-based object detector

Lidar

Lane detector

Inertial measurement unit

Examples of automated driving sensors



Camera (640 x 480 x 3)

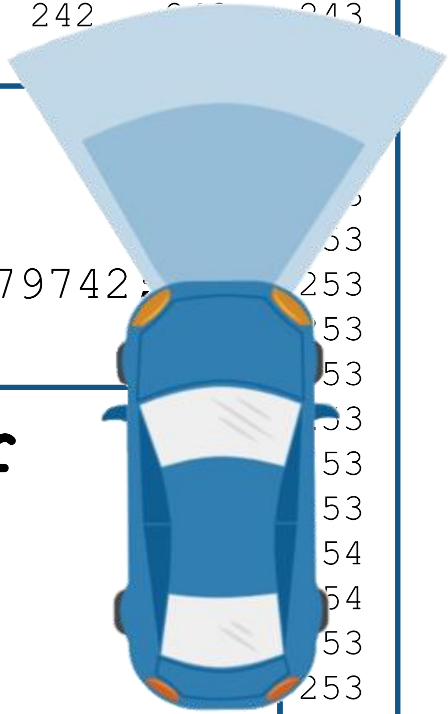
```
239   239   237   238   241   241   241   242         243
252   252   251   252   252   253   253
25
25
25
25                                                     53
25                                                     253
25                                                     53
25                                                     53
25                                                     53
25                                                     53
25                                                     53
25                                                     54
25                                                     54
25                                                     53
25                                                     253
25
25
25
25
25
25
25
```

Vision Detector

```
SensorID      = 1;
Timestamp     = 1461634696379742
NumDetections = 6;
Dete
  Tr
  Cl
  Po
  Ve
  Si
Detec
  Tr
  Cl
  Po
  Ve
  Si
```

Lane Detector

```
Left
  IsValid:        1
  Confidence:     3
  BoundaryType:   3
  Offset:         1.68
  HeadingAngle:   0.002
  Curvature:      0.000
Right
  IsValid:        1
  Confidence:     3
```



Radar Detector

```
SensorID      = 2;
Timestamp     = 1461634696407521;
NumDetections = 23;
Detection
  TrackID
  TrackSt
  Positio
  Velocit
  Amplitu
Detection
  TrackID
  TrackSt
```

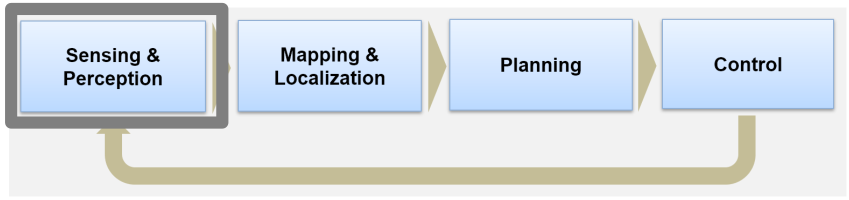
Lidar (47197 x 3)

```
-12.2911    1.4790   -0.5
-14.8852    1.7755   -0.6
-18.8020    2.2231   -0.7
-25.7033    3.0119   -0.9
 -0.0632    0.0815    1.2
 -0.0978    0.0855    1.2
 -0.2814    0.1064    1.2
                      1.2
                      1.2
                      1.2
                      1.2
                     -0.6
                     -0.7
```

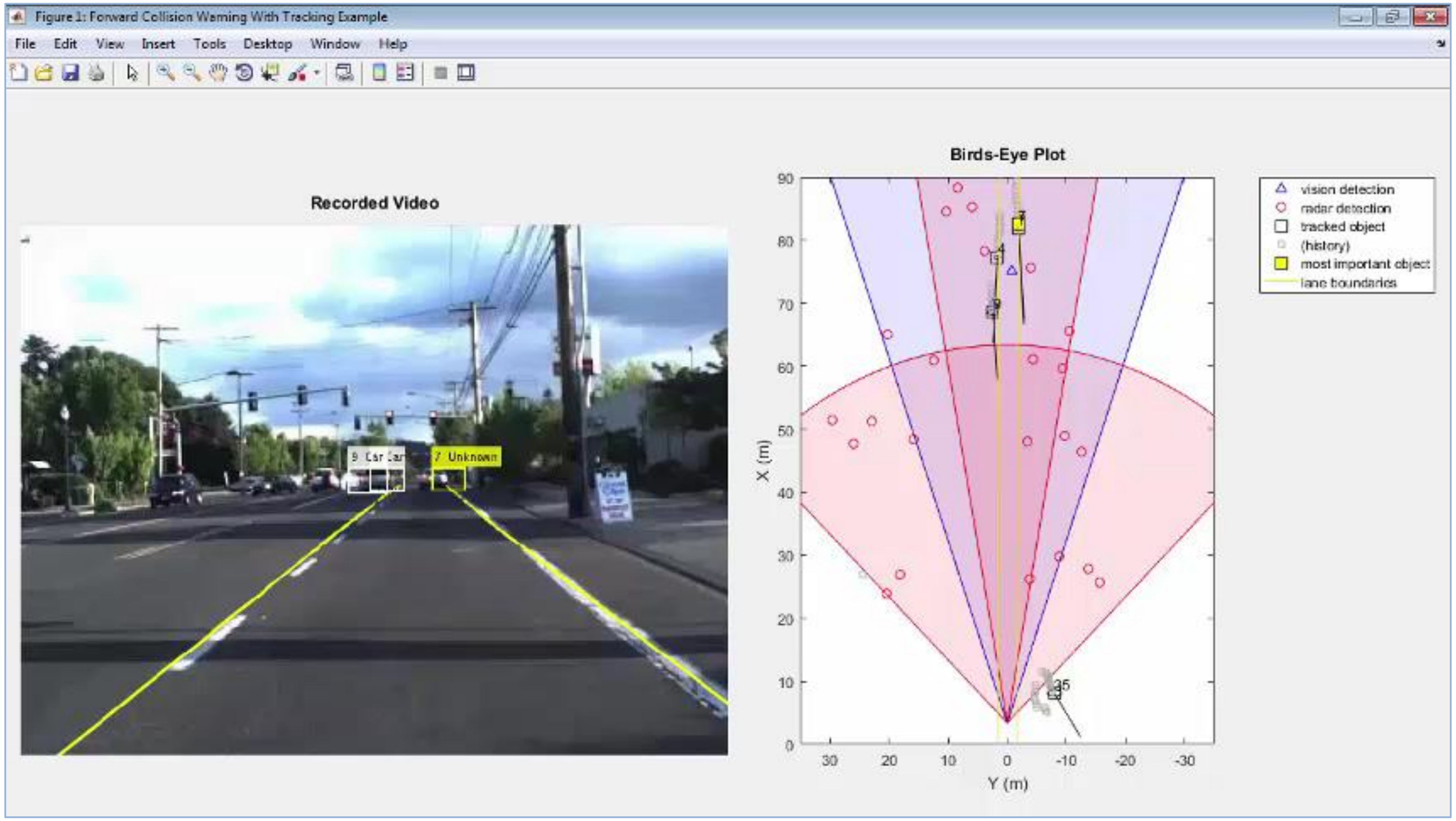
Inertial Measurement Unit

```
Timestamp:   1461634696379742
Velocity:    9.2795
YawRate:     0.0040
```

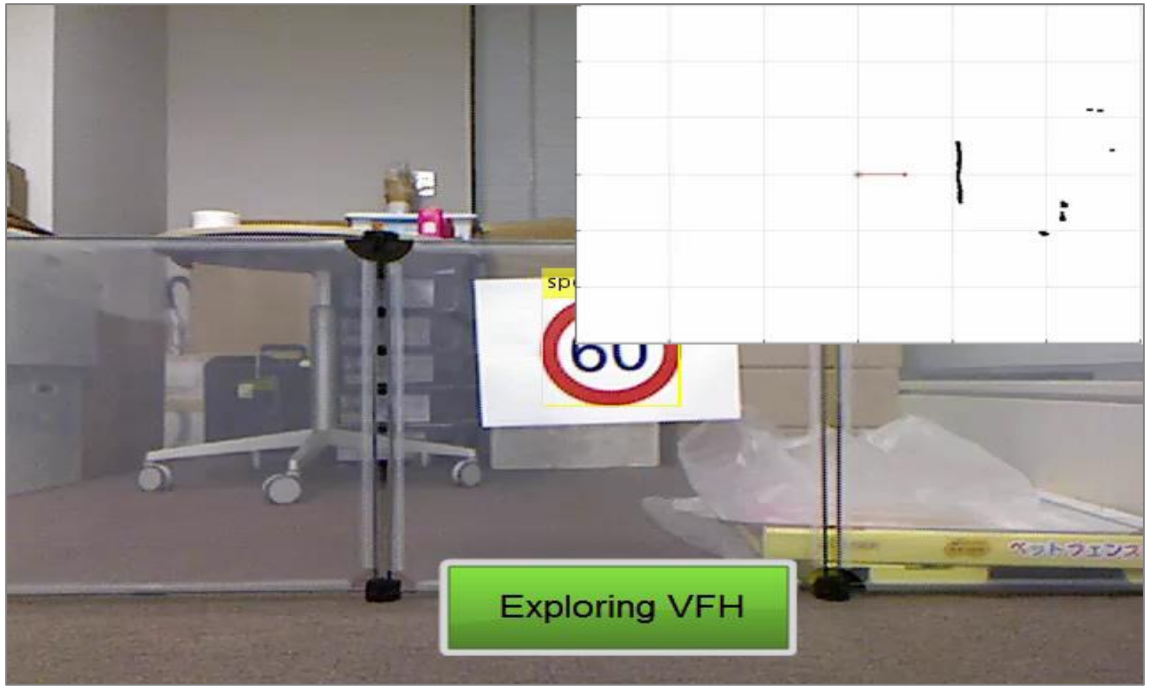
Data visualisation and sensor fusion



- Fusion of data from vision and radar sensors
- Outcome: representation of objects and obstacles around the vehicle
- Basis for a Forward Collision Warning system



Machine Learning & Deep Learning



Object detection and recognition

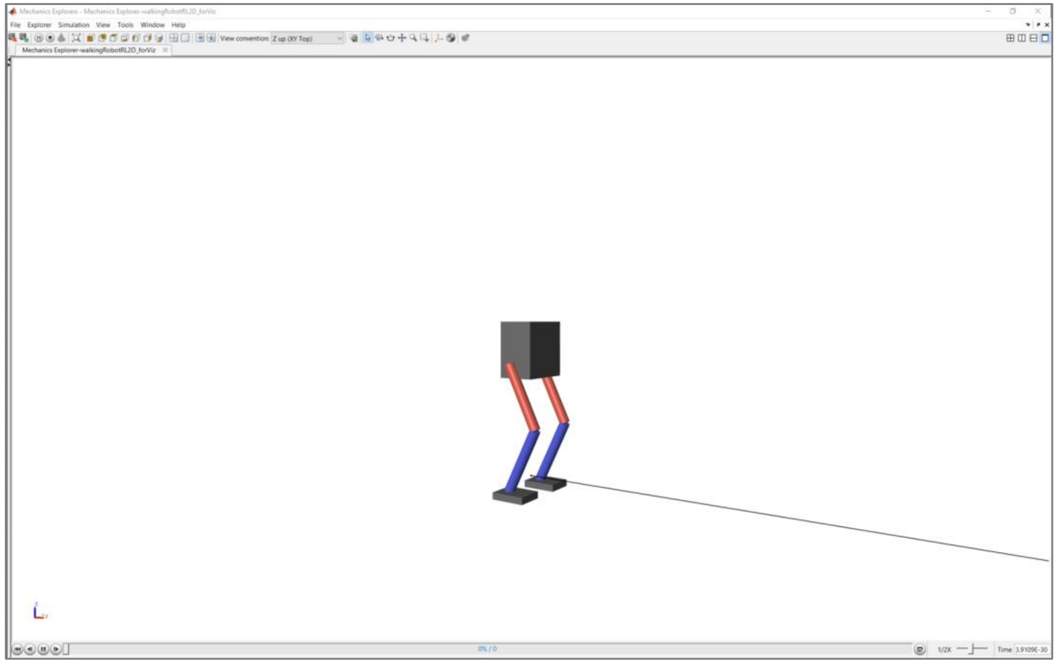
Find traversable terrain using semantic segmentation

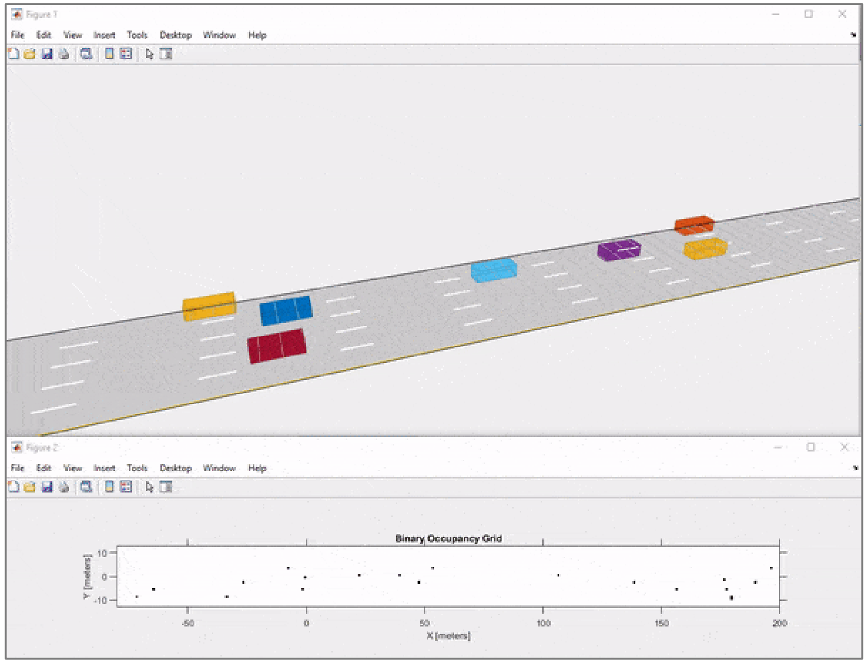
CamVid Dataset

1. Segmentation and Recognition Using Structure from Motion Point Clouds, ECCV 2008
2. Semantic Object Classes in Video: A High-Definition Ground Truth Database, Pattern Recognition Letters

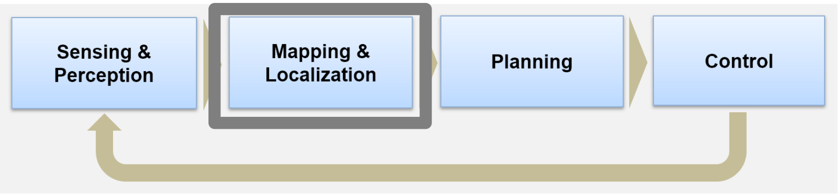
Reinforcement Learning

- What is Reinforcement Learning?
  - Type of machine learning that trains an **‘agent’** through repeated interactions with an environment
- How does it work?
  - Through a trial & error process that uses a reward system to maximize success

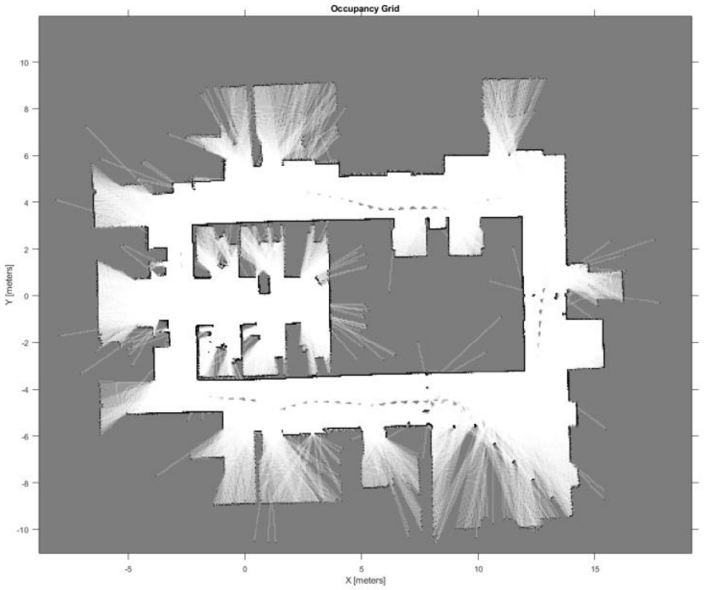


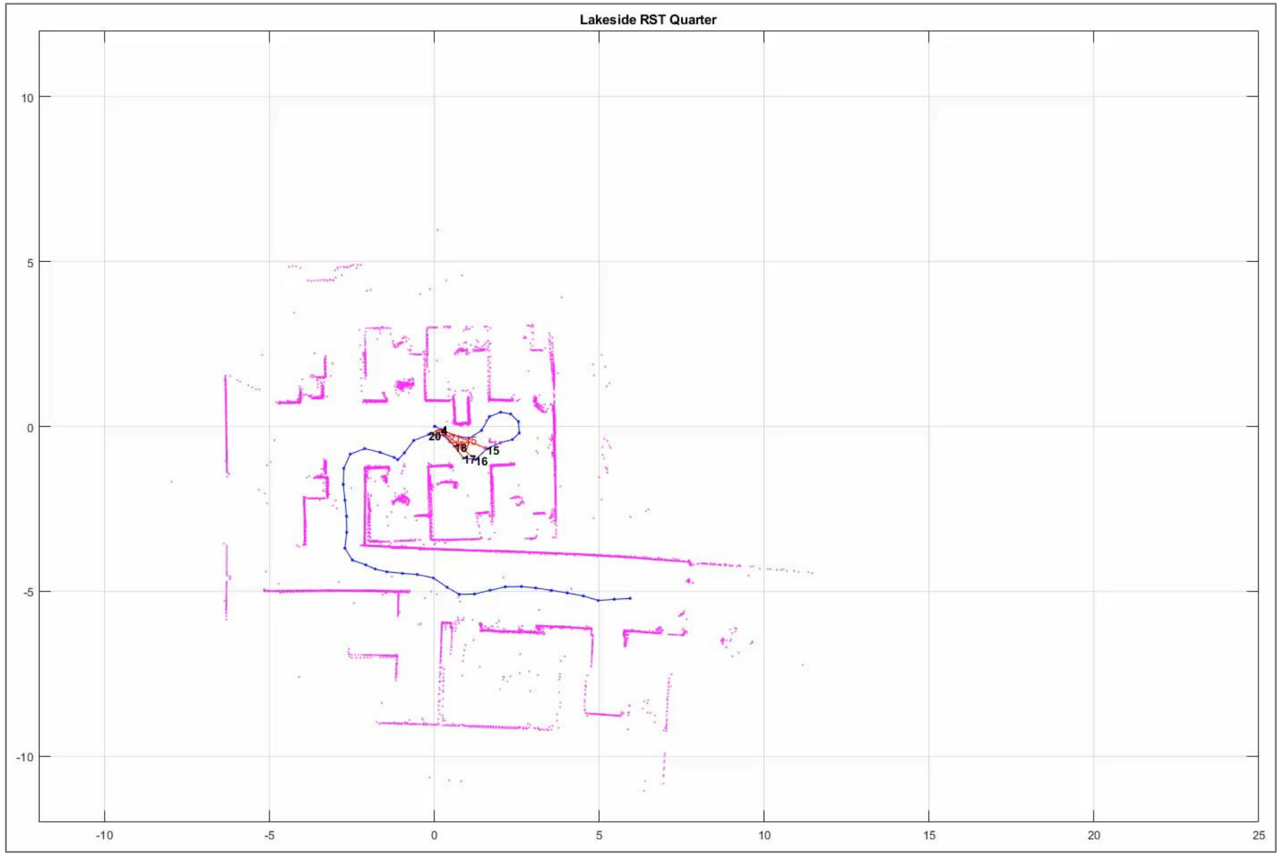


Simultaneous Localisation and Mapping (SLAM)



Expected Map





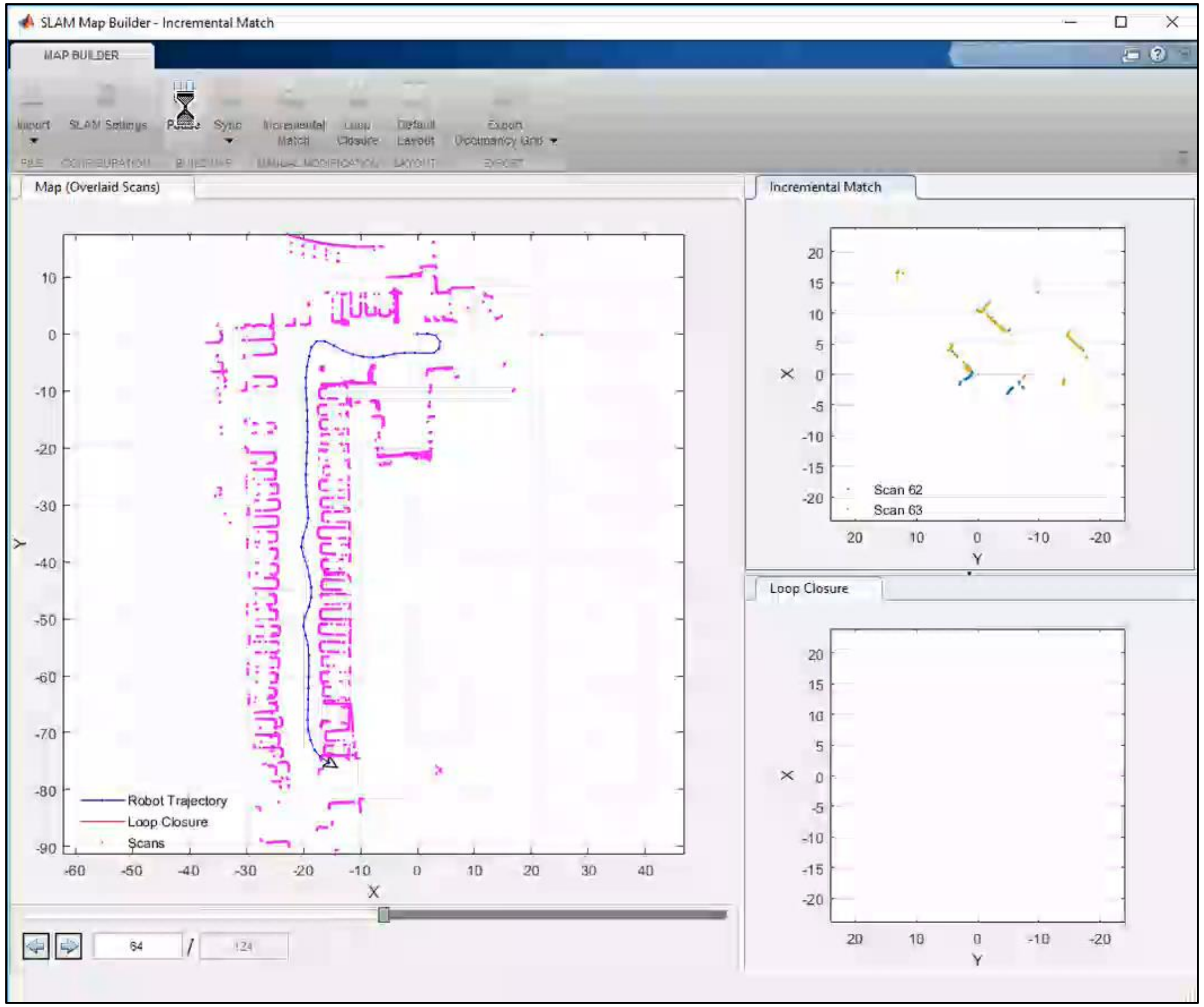
SLAM Map Builder App

- **Build 2D map of environment** based on Lidar and odometry data

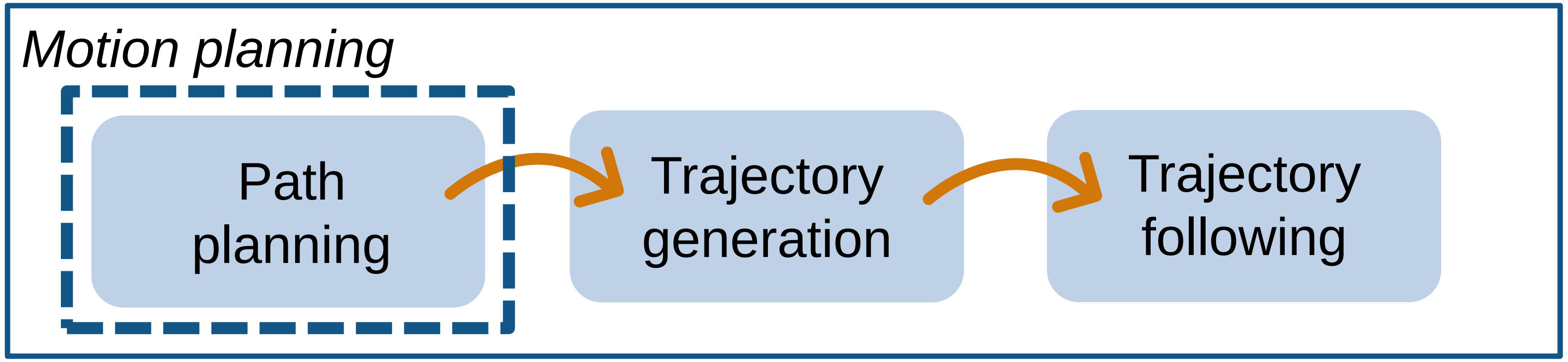
- Modify **loop closures** and **incremental scan matches** to improve map quality

- Export the resulting **occupancy grid** and **use for path planning**

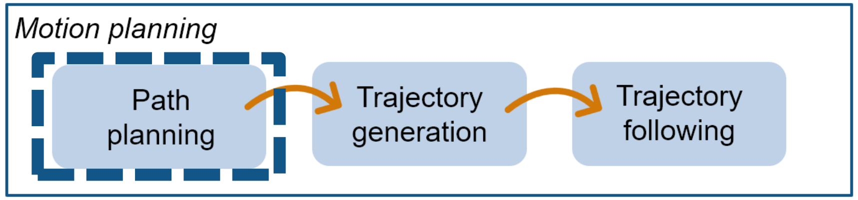
- Loading and saving of sessions to allow saving of work and resuming SLAM map building



Motion planning



Motion planning: path planning



Trajectory Optimization

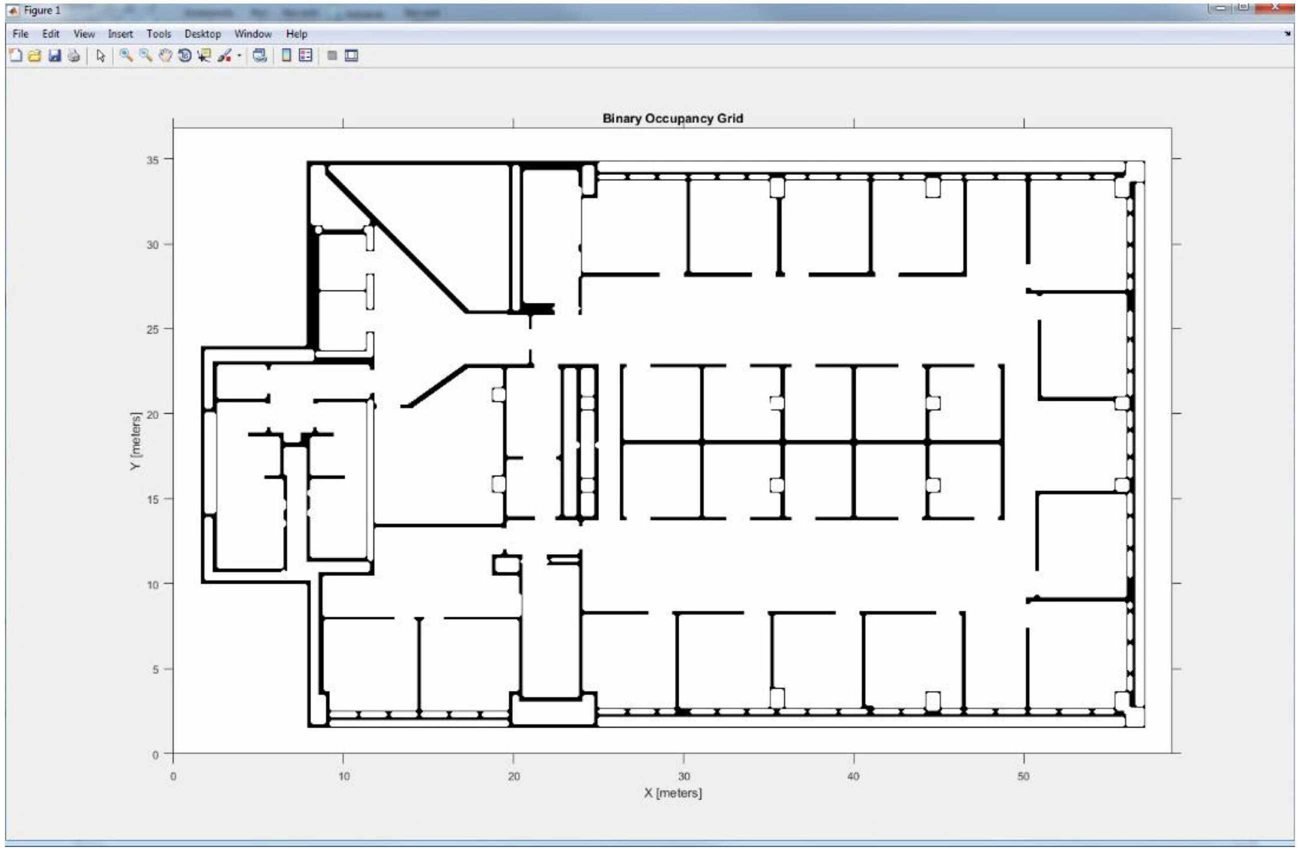
e.g. splines

Grid-based

e.g. Dijkstra’s, A*

Sampling-based

e.g. PRM



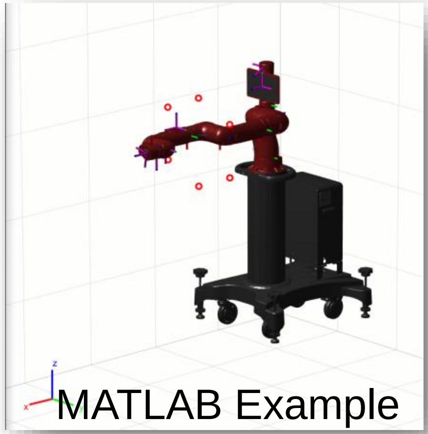
Motion planning: Trajectory Generation

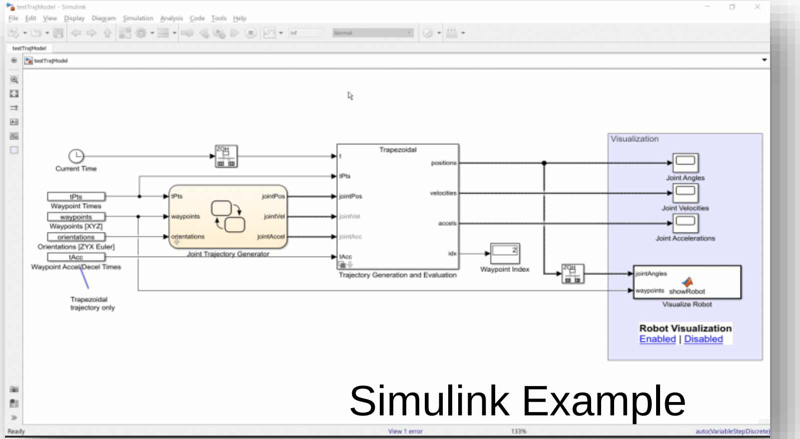
Sensing & Perception → Mapping & Localization → **Planning** → Control

Motion planning

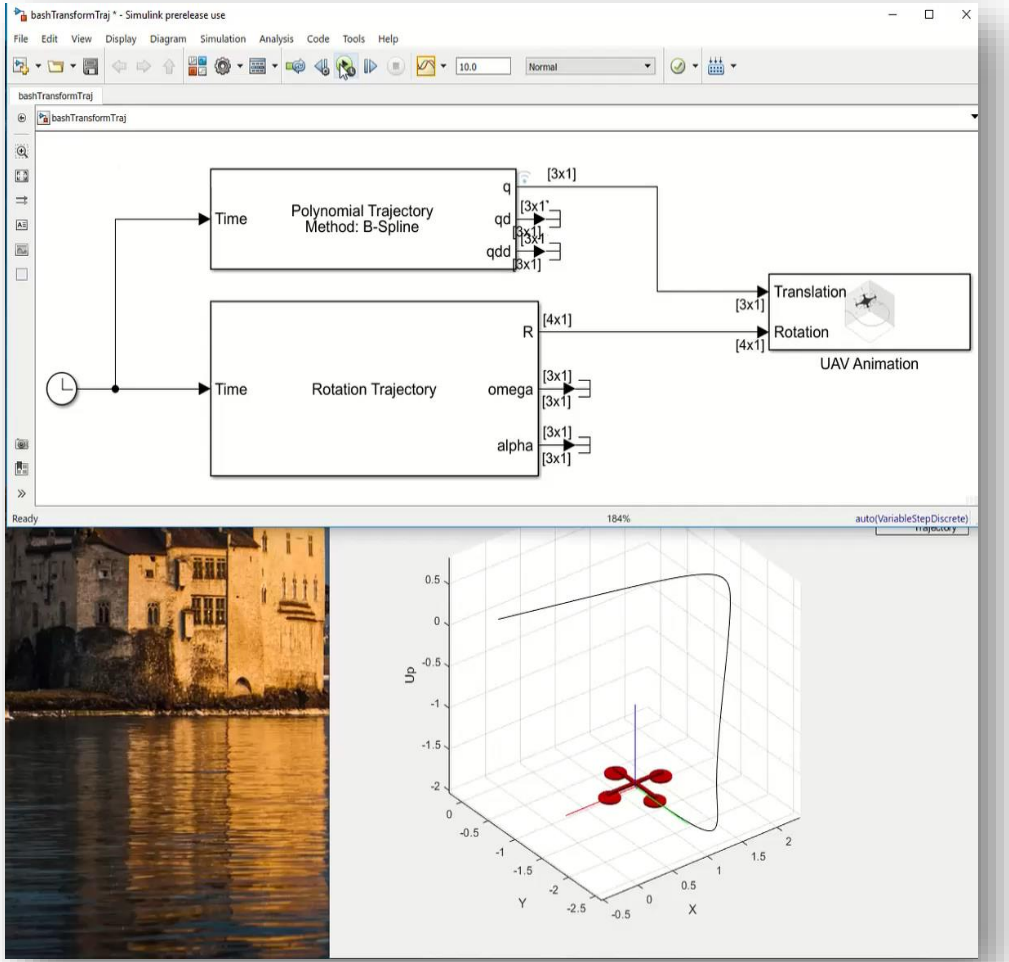
Path planning → **Trajectory generation** → Trajectory following

Use polynomials and splines to generate manipulator trajectories

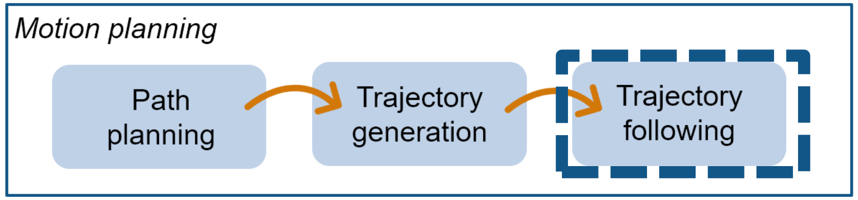




Extensible to other autonomous systems, e.g. UAVs

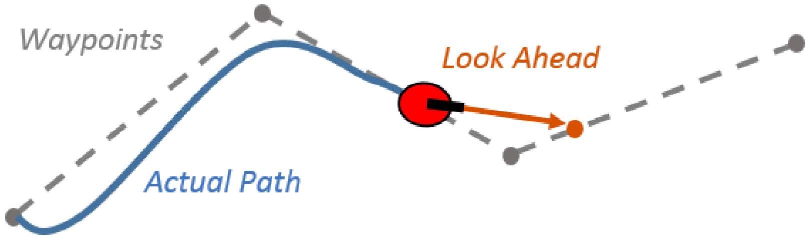


Motion planning: trajectory following

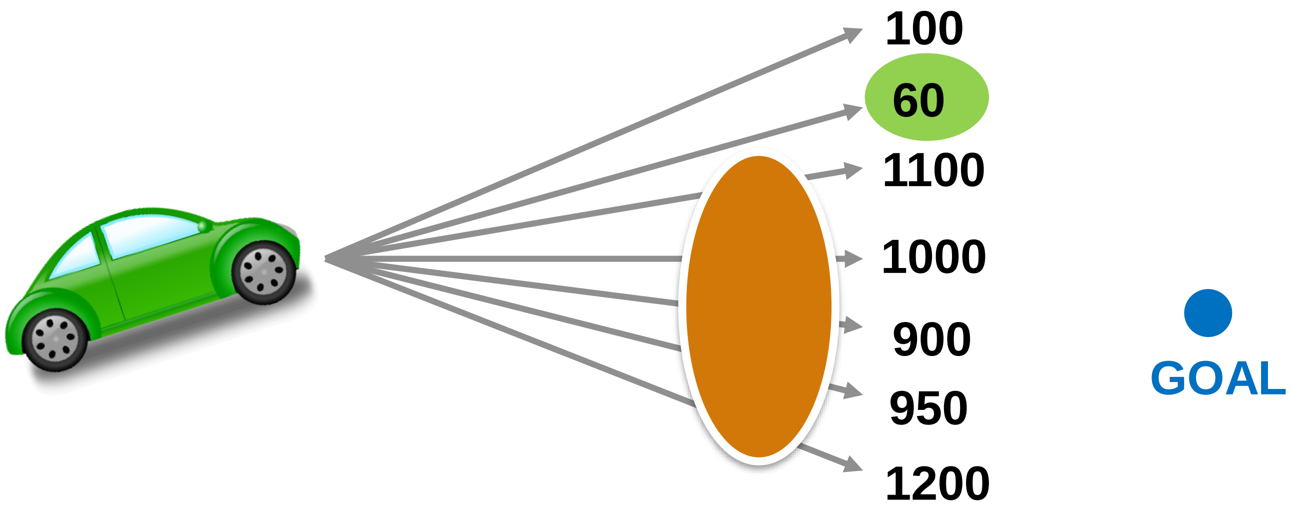


Pure Pursuit

- No collision avoidance!
- Mobile robots or latitude controller for vehicles operating at low speeds



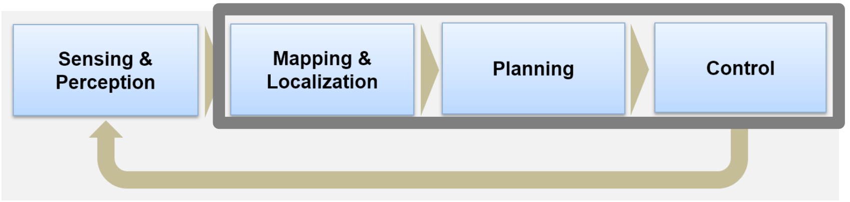
Predictive Control

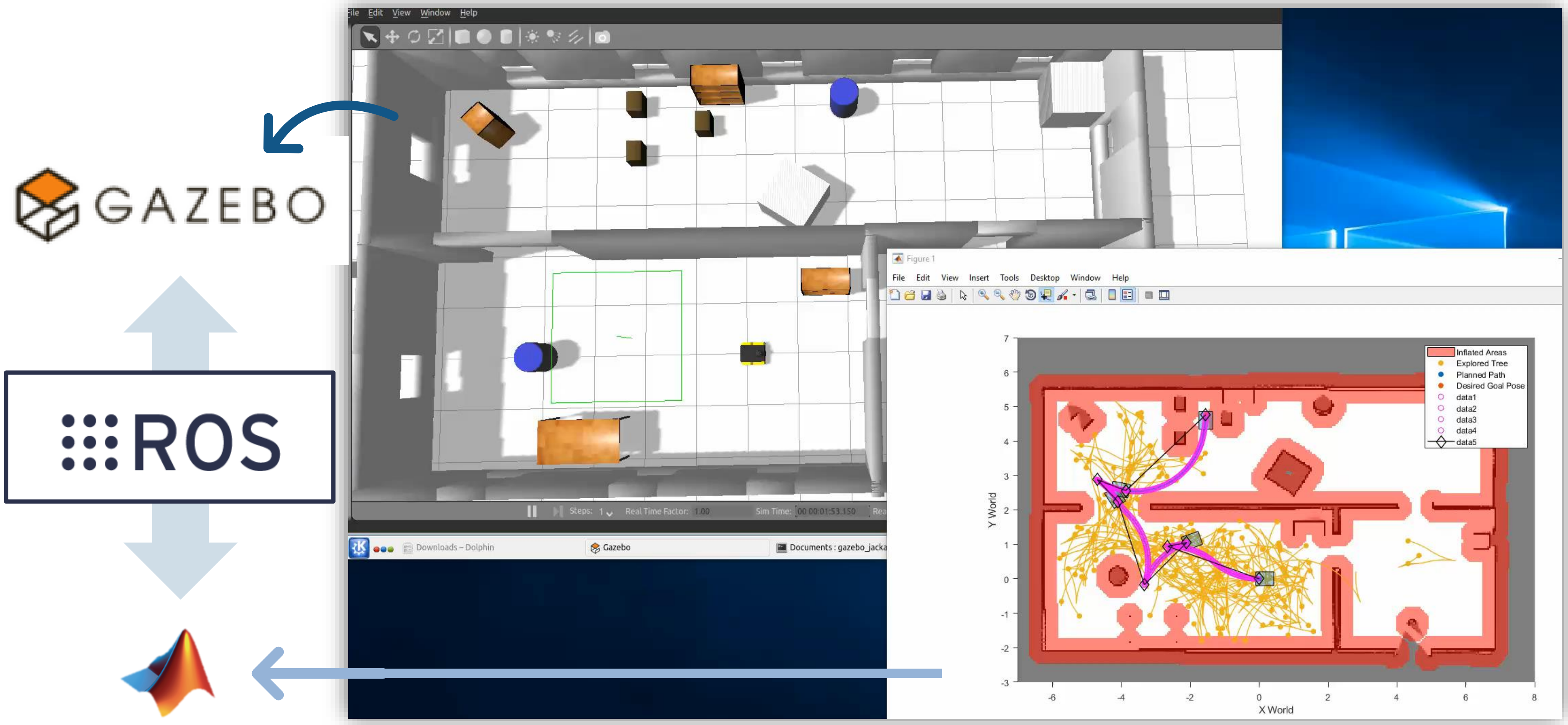


- Rollout all the trajectories
  - Actuator limits / State limits / Motion constraints
- Assign cost to each trajectory
- Select best one and execute

Complete Workflow

SLAM, Planning, Following



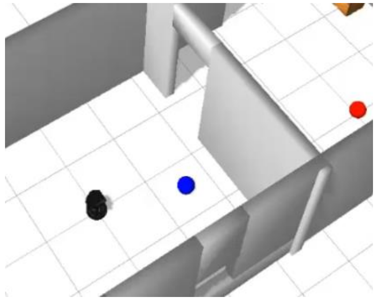


Two ways of interfacing MATLAB/Simulink with ROS

ROS-enabled simulator or robot

Mode 1

ROS Interface



ROS

- Develop algorithm in MATLAB / Simulink
- Algorithm runs on PC

Mode 2

Develop algorithm in Simulink

Generate independent ROS node (C++)

Deploy on robot / ROS network

Robot is autonomous

Design and Test Robotics Algorithms

Test with a physical robot running ROS

Turtlebot at Innorobo
Lyon 2015

Mode 1: Algorithm on PC, communicate with ROS-enabled robot through interface

Mode 2: Independent ROS node on robot

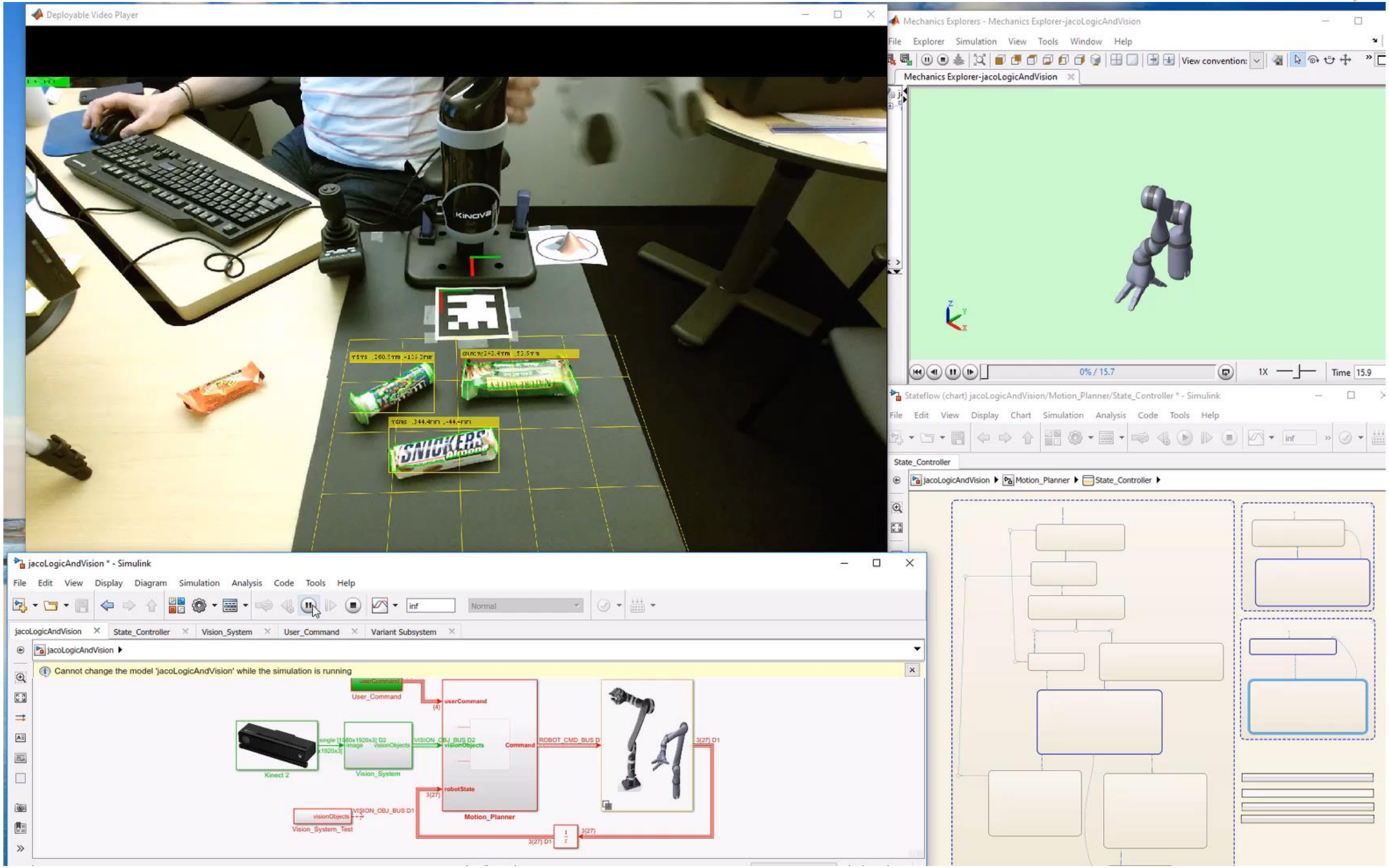
Key Components for Enabling Full Autonomy

Physical Systems & Environment

Algorithms

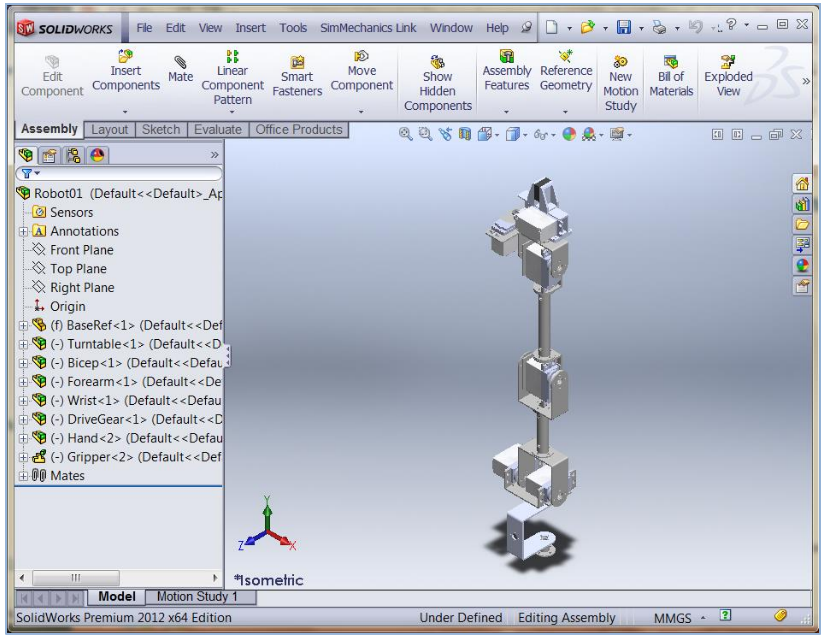
Sensing & Perception → Mapping & Localization → Planning → Control


Manipulators
Ground vehicles
UAVs

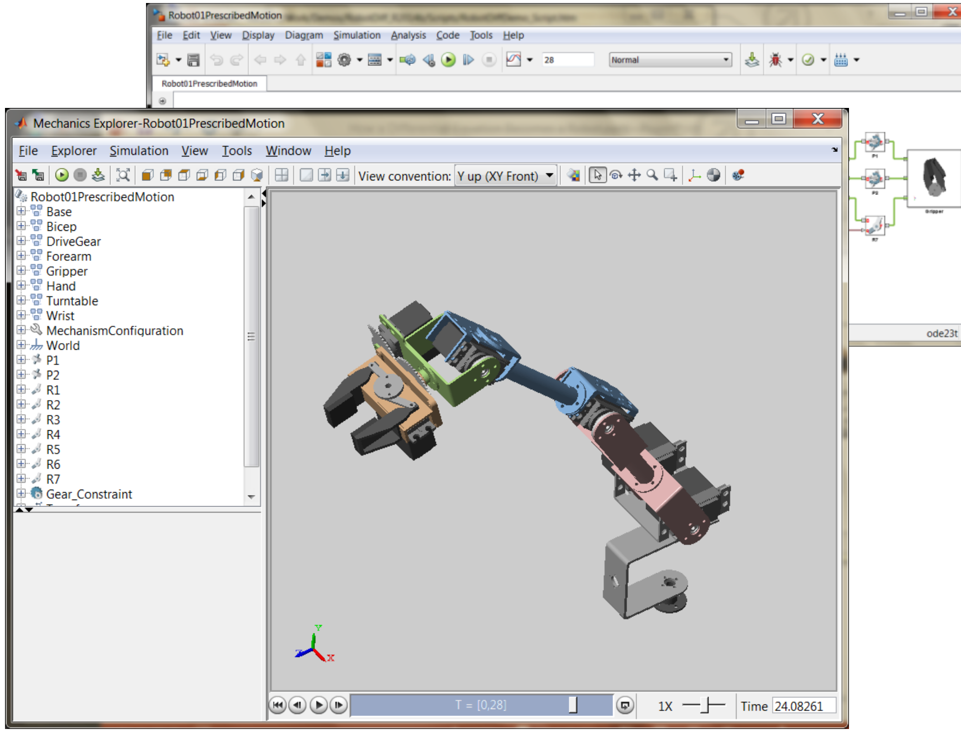

Deployable Video Player
Mechanics Explorers - Mechanics Explorer-jacoLogicAndVision
Mechanics Explorer-jacoLogicAndVision
Stateflow (chart) jacoLogicAndVision/Motion_Planner/State_Controller * - Simulink
State_Controller
jacoLogicAndVision * - Simulink
Cannot change the model 'jacoLogicAndVision' while the simulation is running
Kinect 2
Vision_System
User_Command
Motion_Planner
Vision_System_Test

Mechanics: import models from common CAD tools

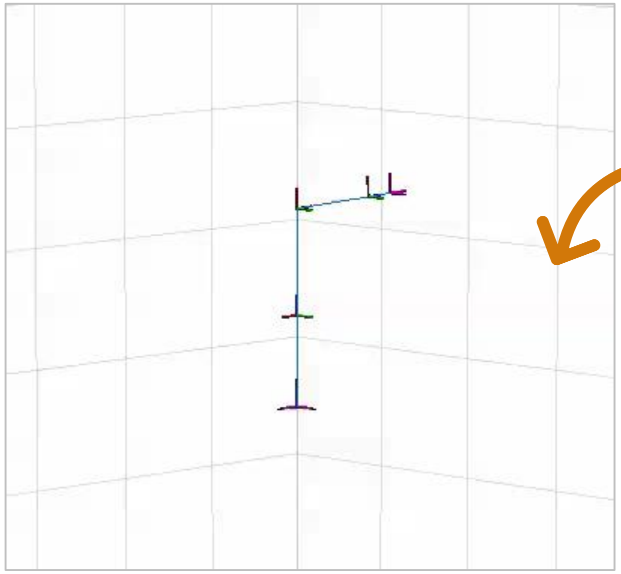
SolidWorks Model



Simscape Multibody Model



Generate rigid body tree from Simscape Multibody models

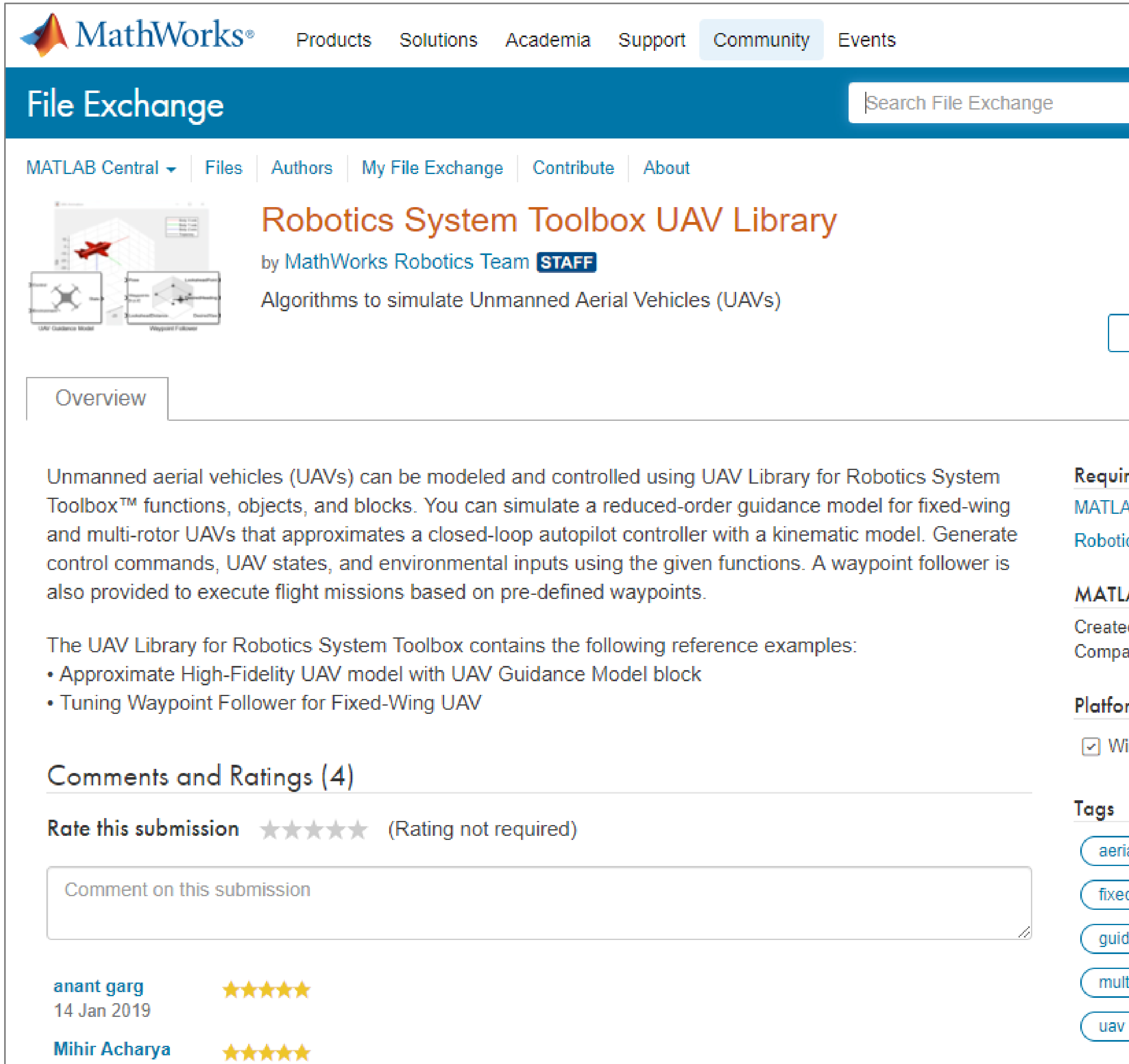


Define rigid body tree and compute dynamics


Manipulators
Ground vehicles
UAVs

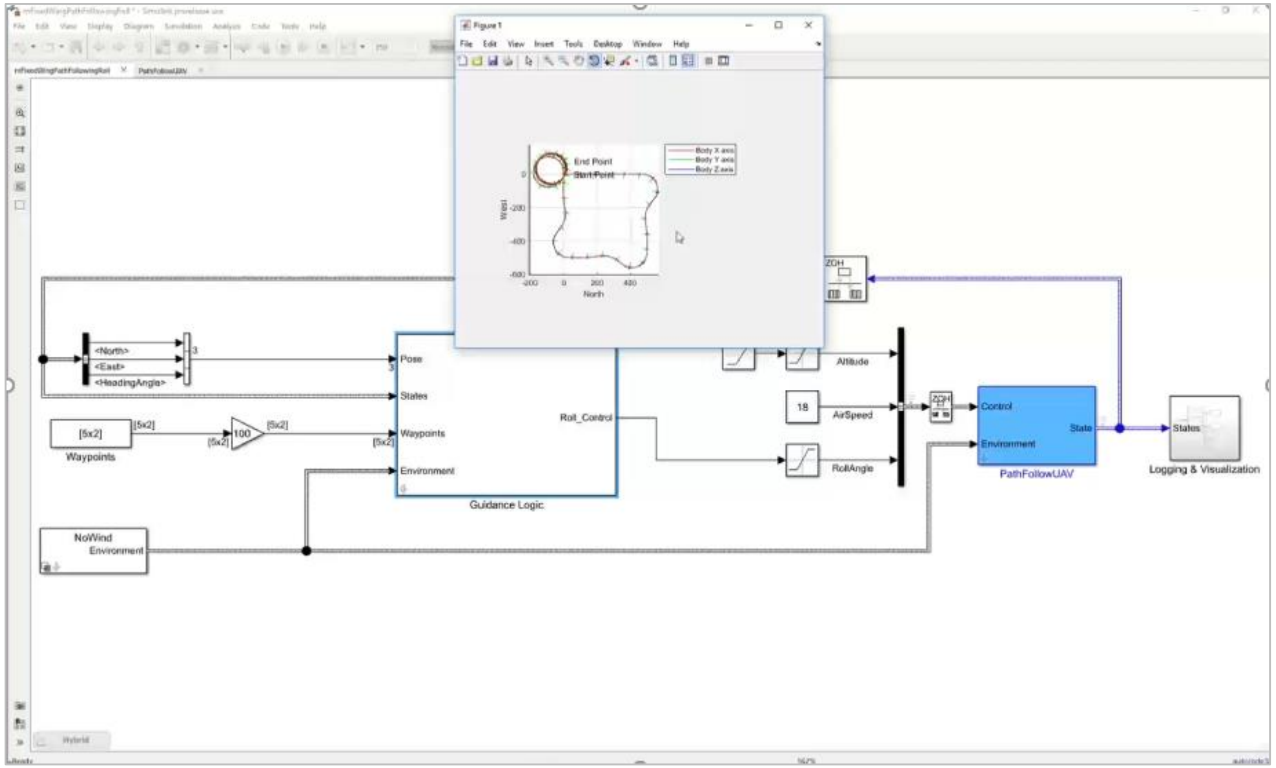
UAV library

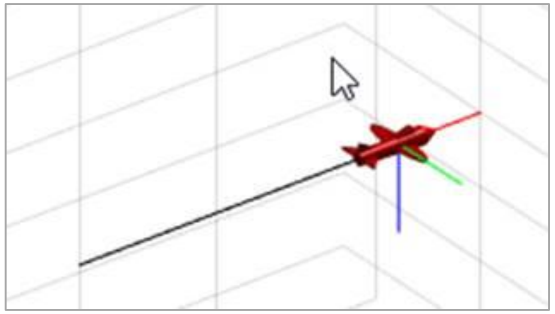
- Support Package for UAVs
- Supports small multi-rotor and fixed-wing aircraft models
- MATLAB functions and Simulink blocks for autonomous flight

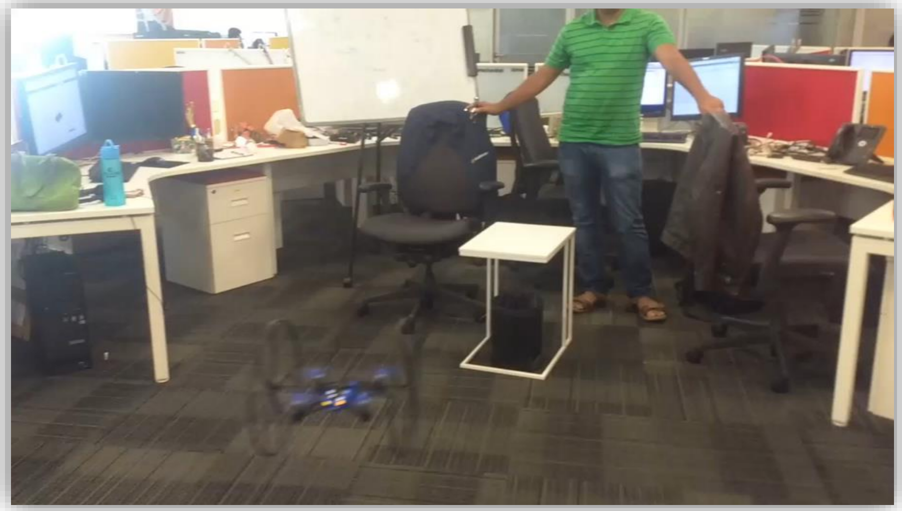


UAV Guidance Block & Waypoint Following

- **Low-fidelity simulation model** a small fixed-wing or multi-rotor UAV with autopilot
- Follow a set of user-defined **waypoints**
- **Deploy algorithm** to UAV hardware







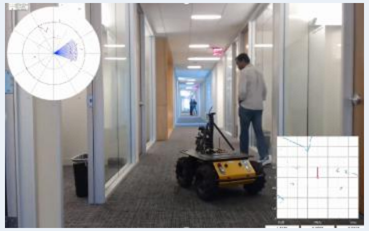
Key Components for Enabling Full Autonomy

Sensing & Perception → Mapping & Localization → Planning → Control

ROS Interface ROS

Deployment to embedded platforms with automatic code generation

C/C++



CUDA

FPGA

Nara Institute of Science and Technology Researchers Develop Tactile Object Recognition Algorithms for a Robotic Hand

NAIST's ROS-enabled Shadow robot performing tactile object recognition.

Challenge

Enable a robotic hand to identify objects based on tactile sensor input

Solution

Use MATLAB to develop machine learning and object recognition algorithms and use Robotics System Toolbox to establish connectivity between the algorithms and the ROS-enabled robot

Results

- Hundreds of manual steps eliminated
- Opportunities to try new algorithm ideas increased
- Expertise shared with other students and researchers

"With Robotics System Toolbox, we seamlessly connected to and controlled our robot directly from the algorithms we had developed in MATLAB, enabling us to minimize development time. We used the time we gained to further our research into new tactile object recognition algorithms."

Takamitsu Matsubara

Nara Institute of Science and Technology

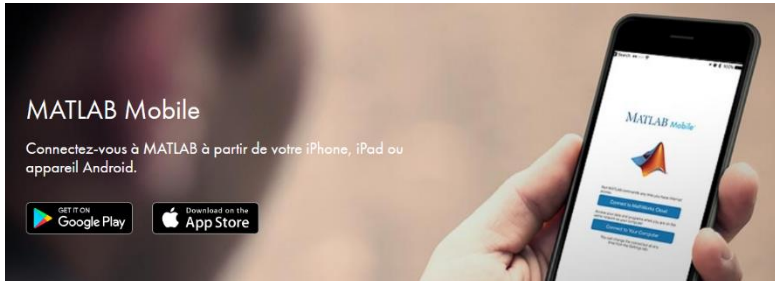
Link to user story

Supports Pédagogiques

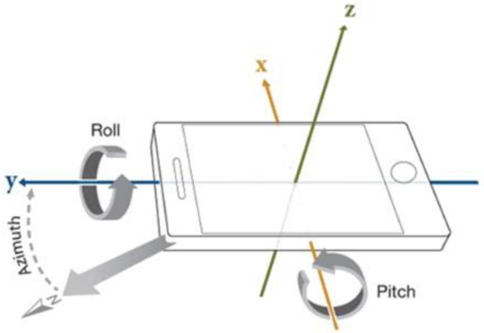
Ninon Candalh-Touta
Ingénieure Pédagogique

Outils Pédagogique

➢ Projets interactifs à bas coût



- Connexion au Cloud
- Connexion à votre ordinateur
- Collecter des données de capteurs



- Accélération sur 3axes
- Vitesse angulaire sur 3 axes
- Champ magnétique sur 3 axes
- Orientation (azimut, tangage et roulis)
- Position (latitude, longitude, altitude, précision horizontale, vitesse, course)

[See examples](#)

MathWorks Supported Hardware

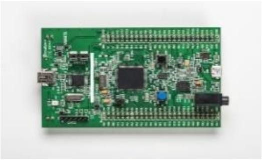
Arduino

Raspberry Pi

Lego EV3

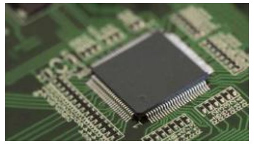
Kinect for Windows

BeagleBone



STM Electronics

Zynq SDR



Texas Instruments

Arduino Kit

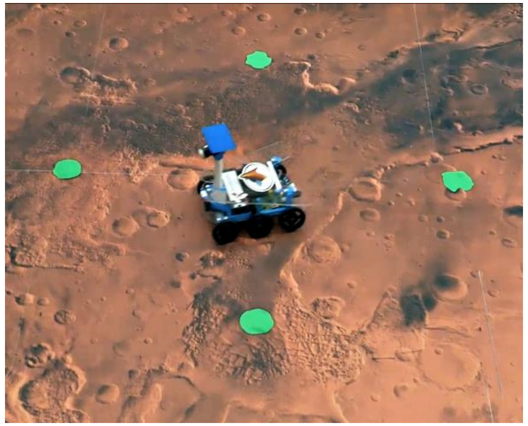
Parrot Mini-Drone

Outils Pédagogique

- MathWorks Rover Robot: a platform for multidisciplinary project

MathWorks

[Rover Robot Competition]



Véhicule Autonome

CentraleSupélec

RENAULT

Student's point of view

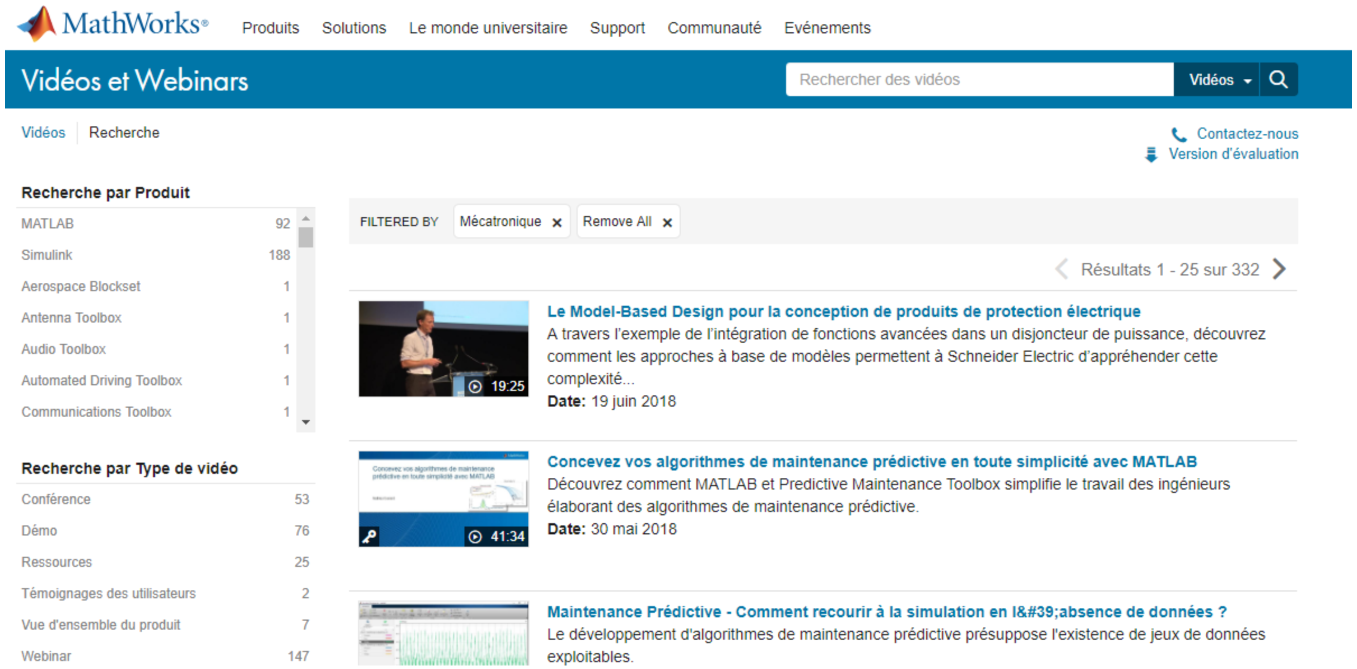
"The MathWorks rover is an **inspiring** and **exciting** platform to work on. It is a concrete project with real constraints. It enabled us to

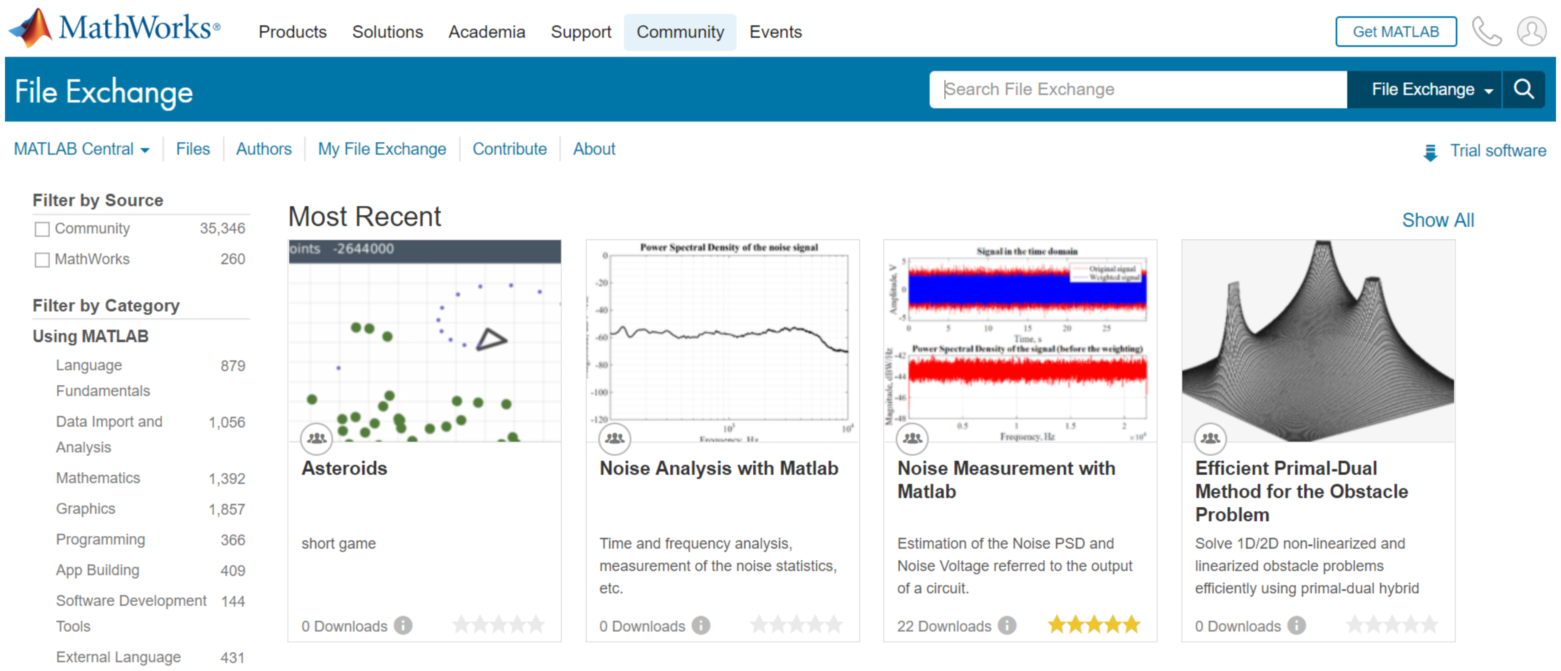
- see the **difference between simulation** and **implementation** on a real system
- show the interest of all the knowledge we learn in class and **allow us to mix them up**.

It was for us a **fun way** to practice with the MATLAB/Simulink software suite. "

Outils Pédagogique

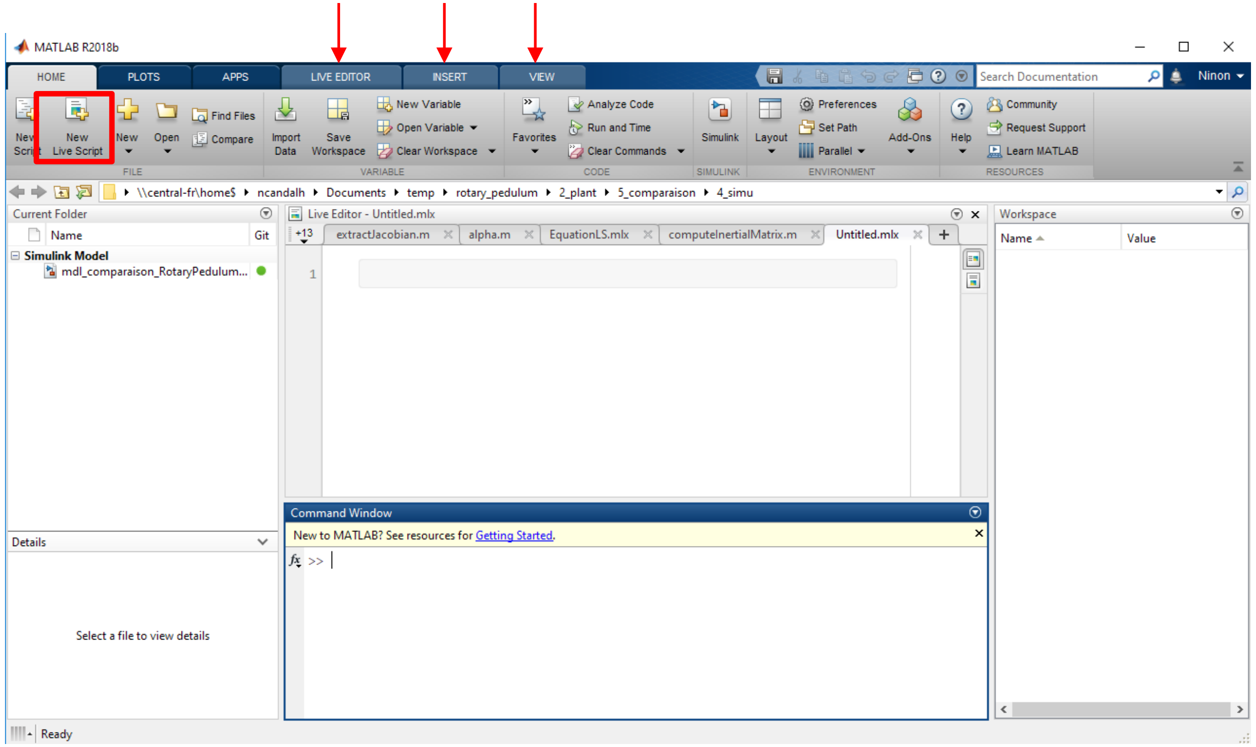
- ➢ Cours et Exemples en ligne

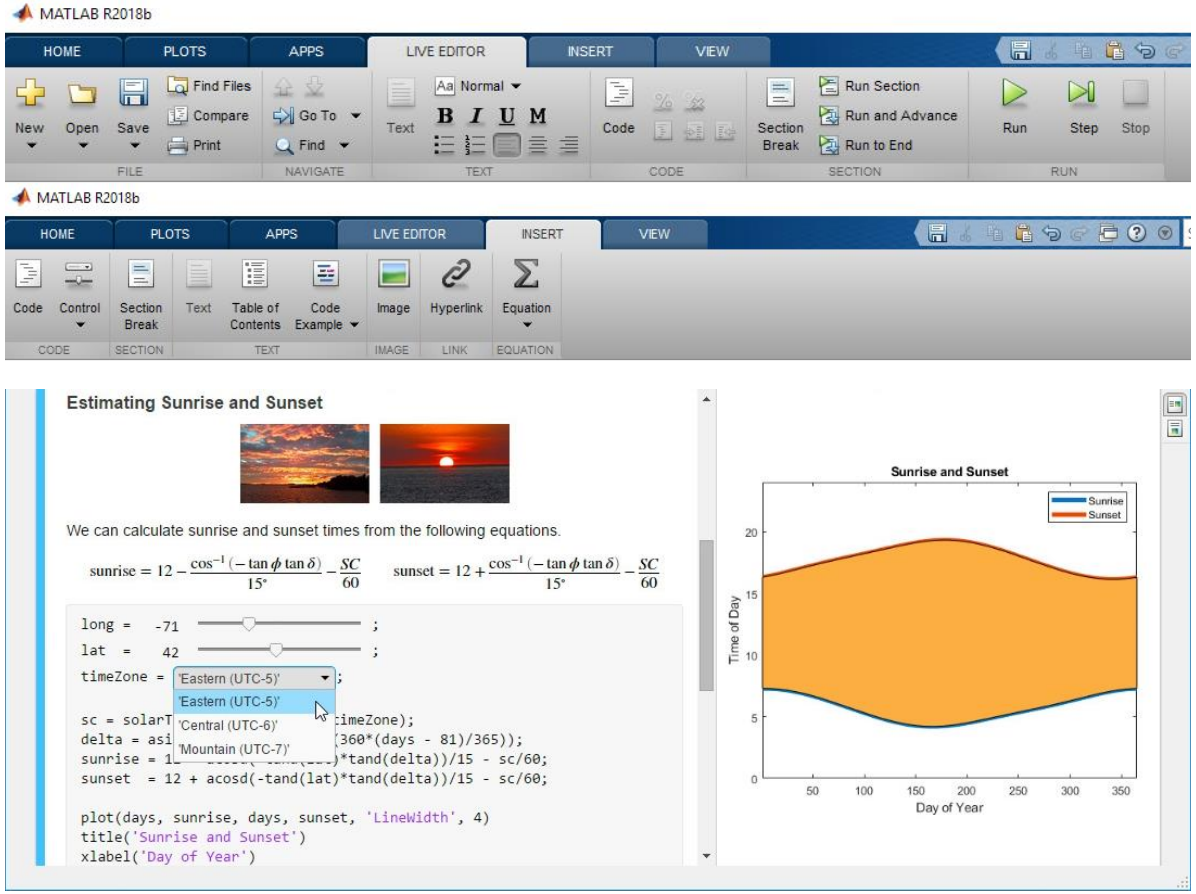




Outils Pédagogique

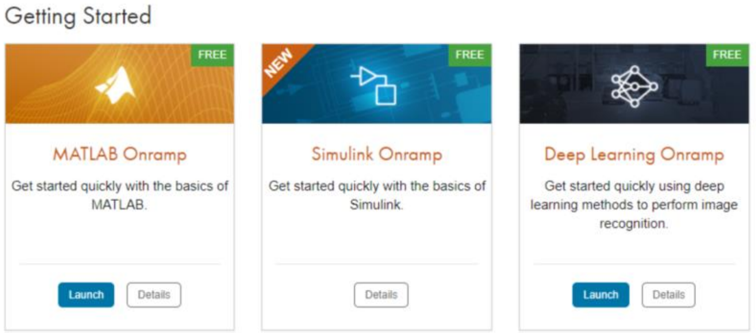
- ➢ [Live Editor]

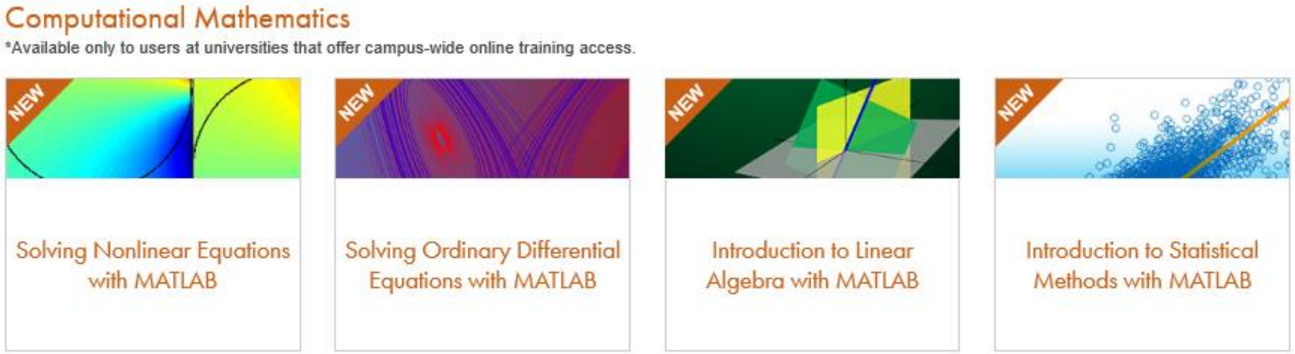




Outils Pédagogique

➢ Formations en ligne



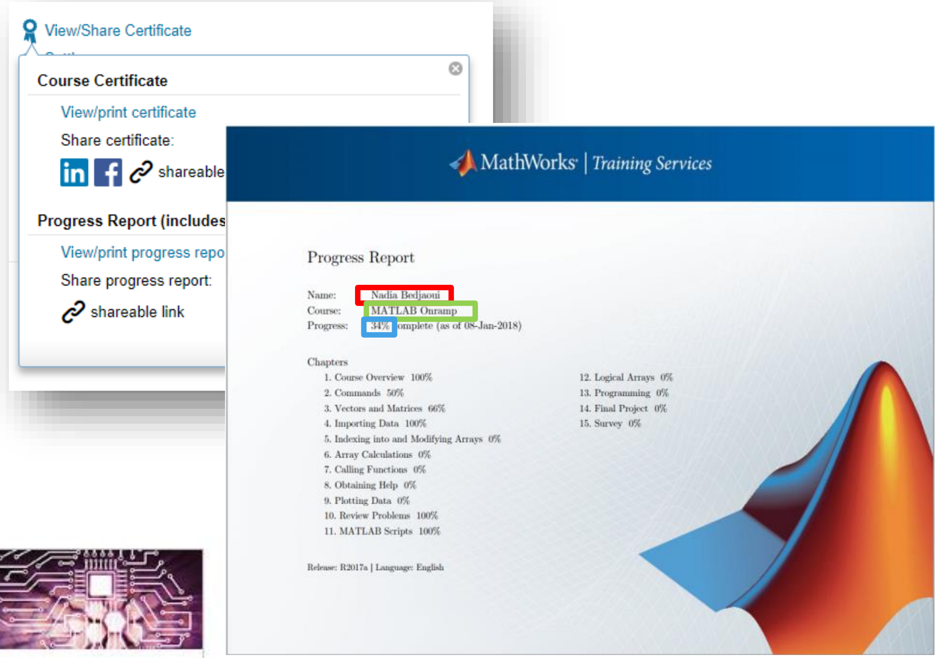


UserStory

@CentraleMarseille

Cours en autonomie
260 sur 263 étudiant(e)s ont validé

+ Retours positifs des autres enseignants, les élèves sont meilleurs en MATLAB et sont plus autonomes
+ Plus de temps « perdu » dans l'apprentissage des fondamentaux de MATLAB



Merci,
avez-vous des questions?

Venez discutez au Stand MathWorks