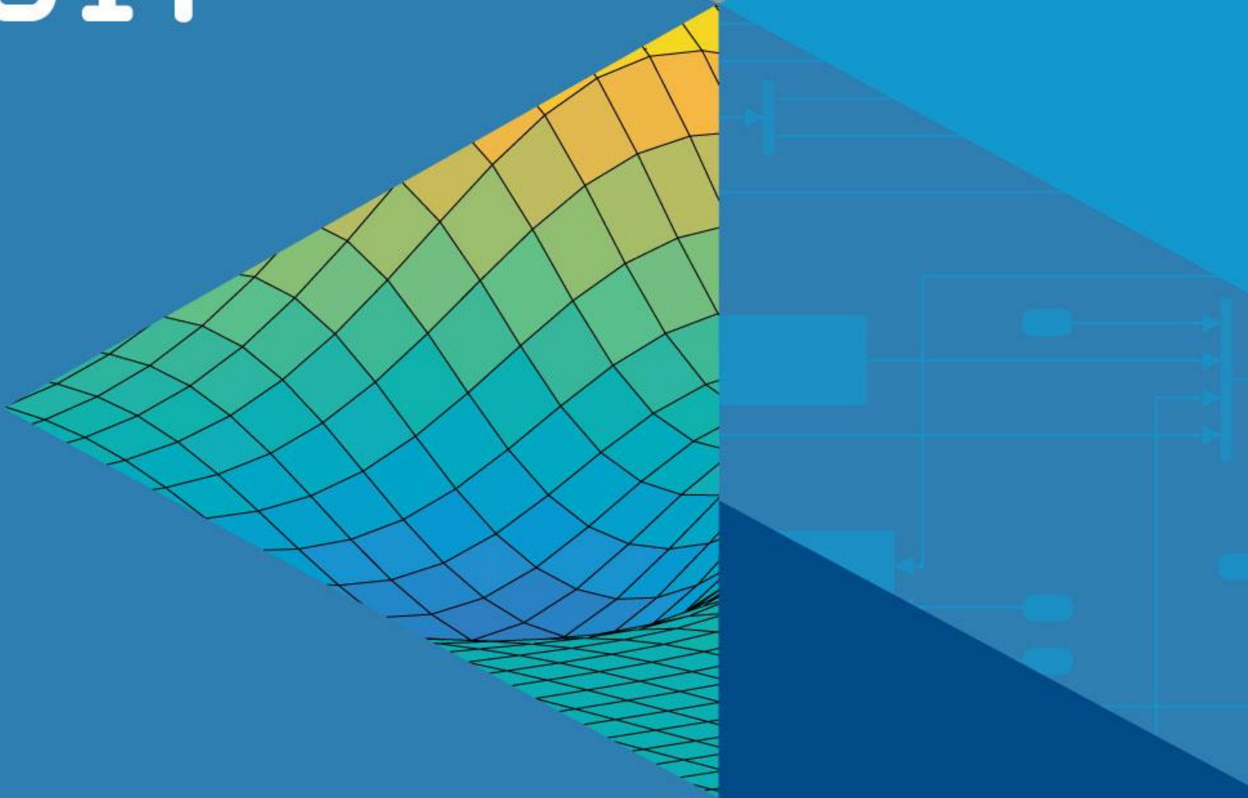
MATLAB EXPO 2017
KOREA

4월 27일, 서울

등록 하기 matlabexpo.co.kr



다중 센서 기반 자율시스템의 모델 설계 및 개발

이제훈 차장

What we will see today…

Functional Segmentation of Autonomous System

Aircraft/ Platform

- Control Surfaces, slats, flaps
- Lifting Body
- Landing Gear
- Battery
- Power Management

Sense

- Radar
- Camera
- Lidar
- EO/IR
- IMU
- GPS-INS
- HW Certification

Perceive

- Environment mapping
- Classification
- Segmentation
- Object Detection
- Sensor Fusion

Plan & Decide

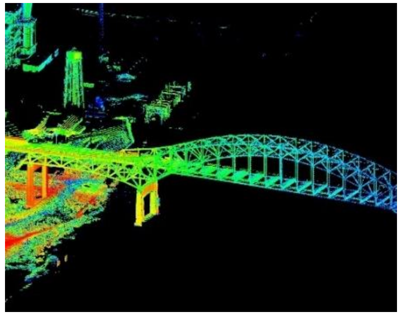
- Object Avoidance
- Path & motion planning
- SLAM

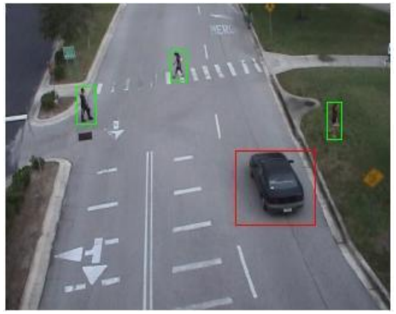
Control

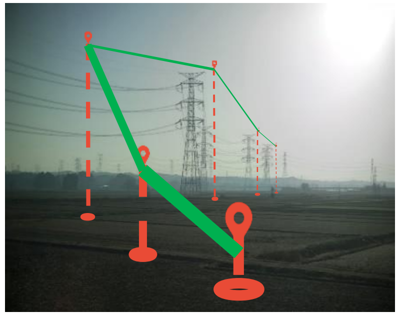
- Guidance, Navigation & Control
- Flight SW certification

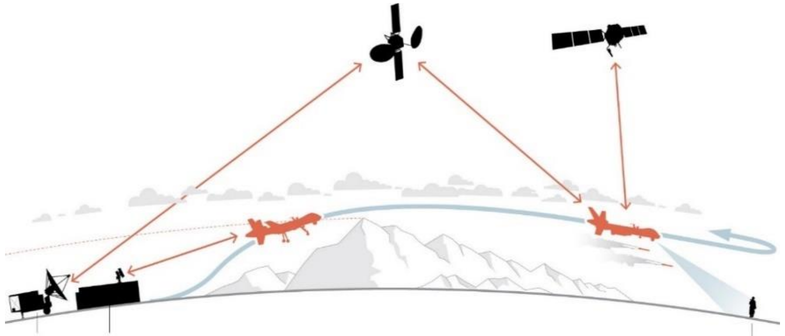
Connect/ Communicate

- Communication with ground operator
- Multi-agent communication
- Satellite data link









Functional Segmentation of Autonomous System

Customer Groups

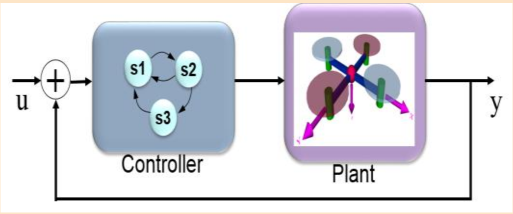
"Autonomous Algorithms"

Aircraft	Perception		Planning		Communication
Platform	**Sense**	**Perceive**	**Plan & Decide**	**Control**	**Connect/ Communicate**
• Control Surfaces, slats, flaps • Lifting Body • Landing Gear • Battery • Power Management	• Radar • Camera • Lidar • EO/IR • IMU • GPS-INS • HW Certification	• Environment mapping • Classification • Segmentation • Object Detection • Sensor Fusion	• Object Avoidance • Path & motion planning • SLAM	• Guidance, Navigation & Control • Flight SW certification	• Communication with ground operator • Multi-agent communication • Satellite data link

Autonomous System Development Workflow

Aerodynamics and Flight Control → Autonomous algorithm → Test and Refine in Simulation → Test and Refine on Real Robot

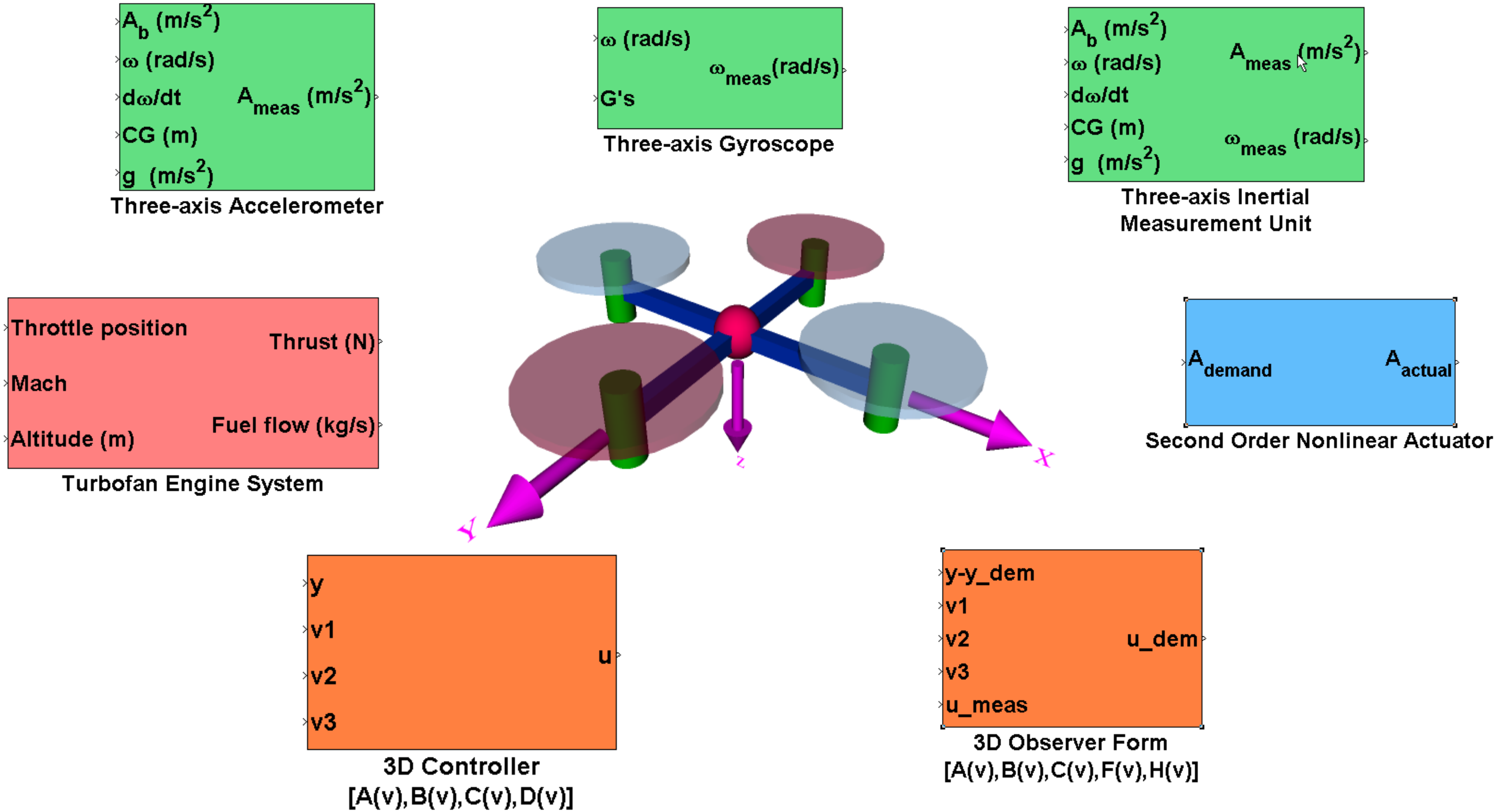
Challenge 1:
Understand
the dynamics
with control algorithm



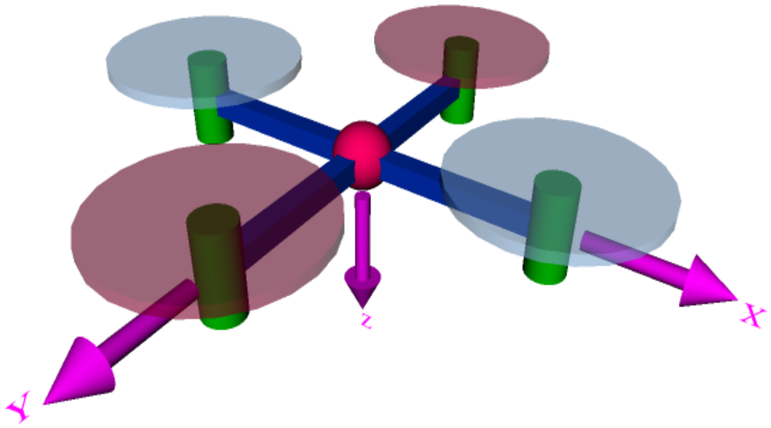
Challenge 2:
Design and Verify
autonomous
algorithms

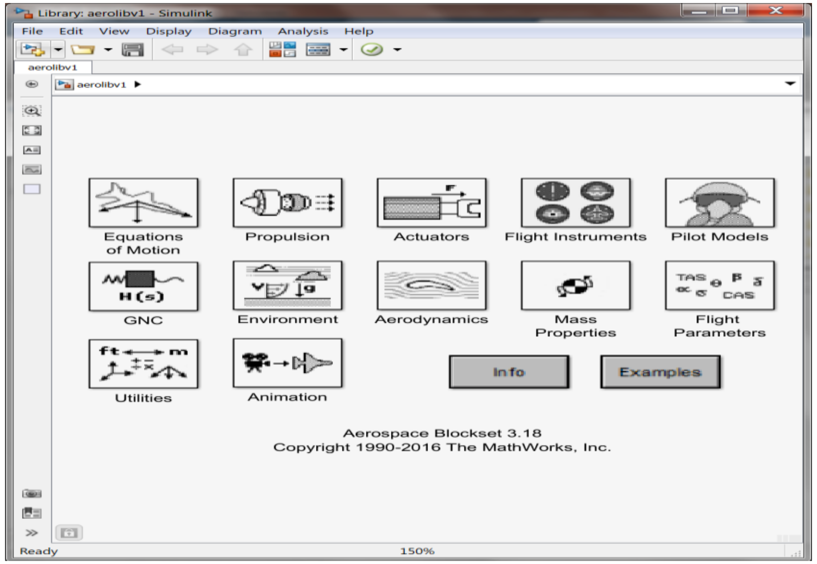
Challenge 3:
Verify and Implement
the algorithm
on to a real hardware

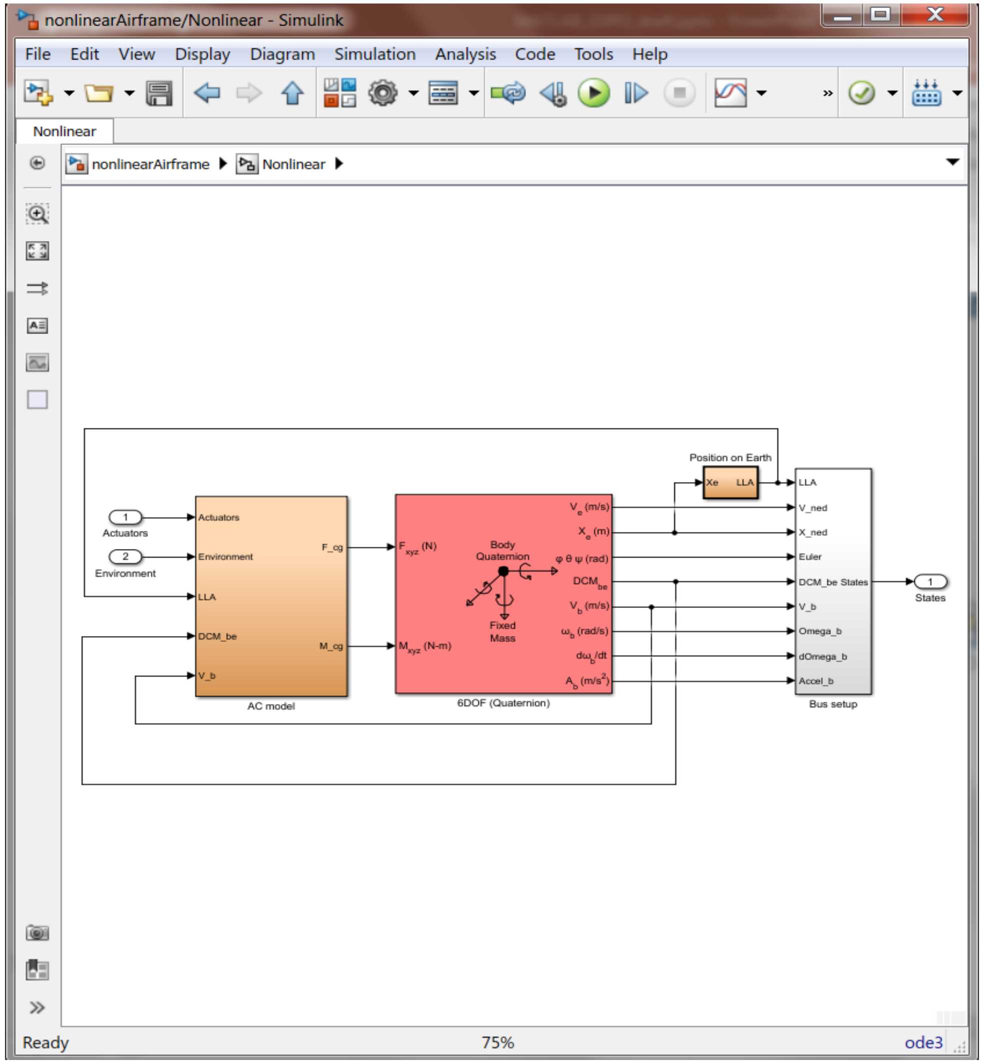
Design **Control algorithm** with Dynamics



Design **Control algorithm** with Dynamics





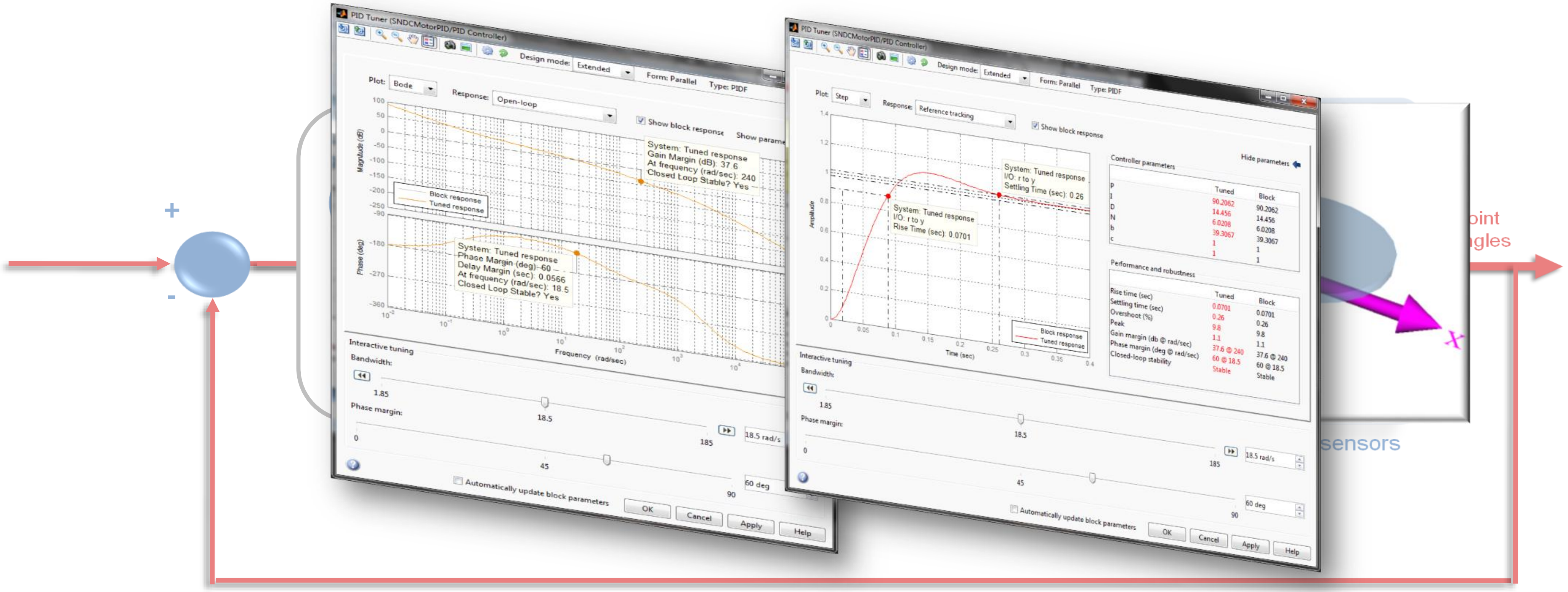


Design **Control algorithm** with Dynamics

Simulink Control Design & Control System Toolbox

Simulating plant and controller ***in one environment***

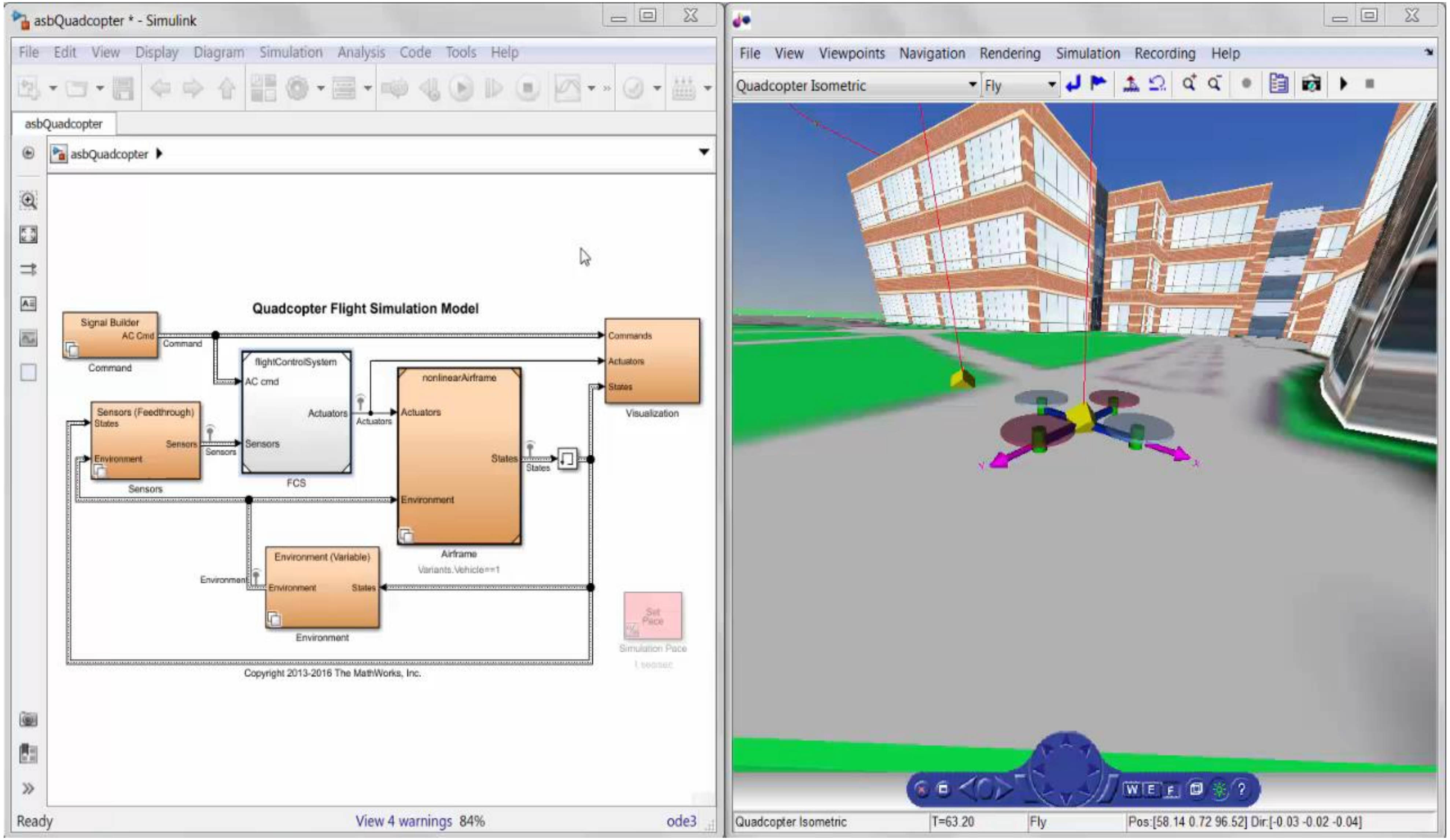
Optimize system-level performance & Closed-loop simulation



Simulink Control Design & ***Control System Toolbox***

automatically linearize the plant, design and tune your PID controllers

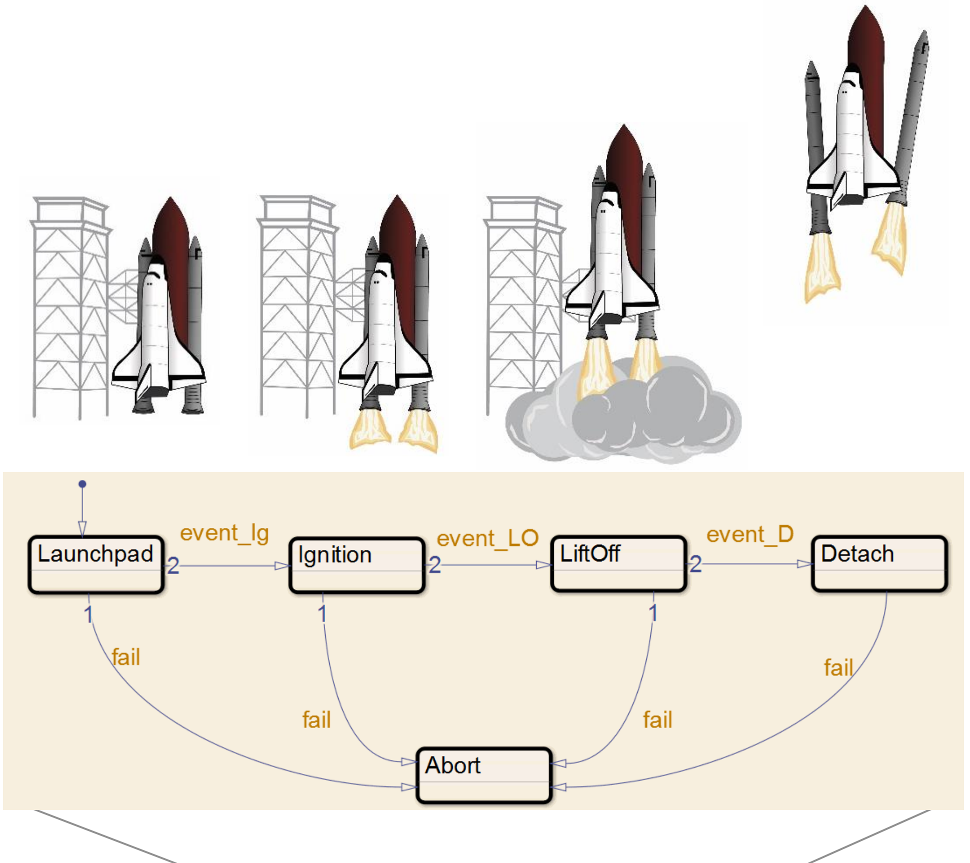
Design **Control algorithm** with Dynamics

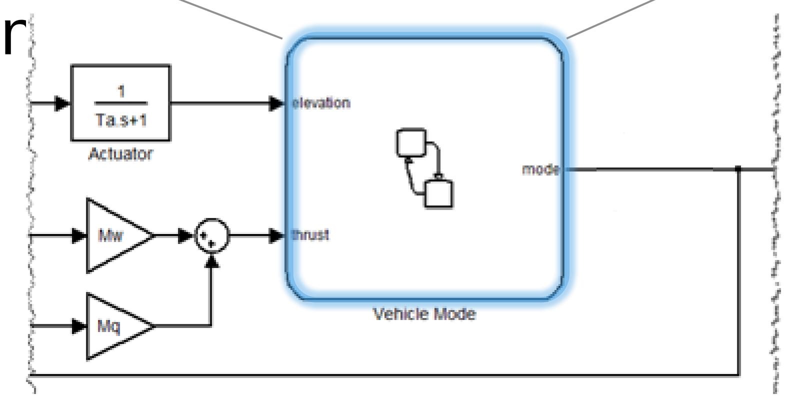


Design **Control algorithm** with Dynamics

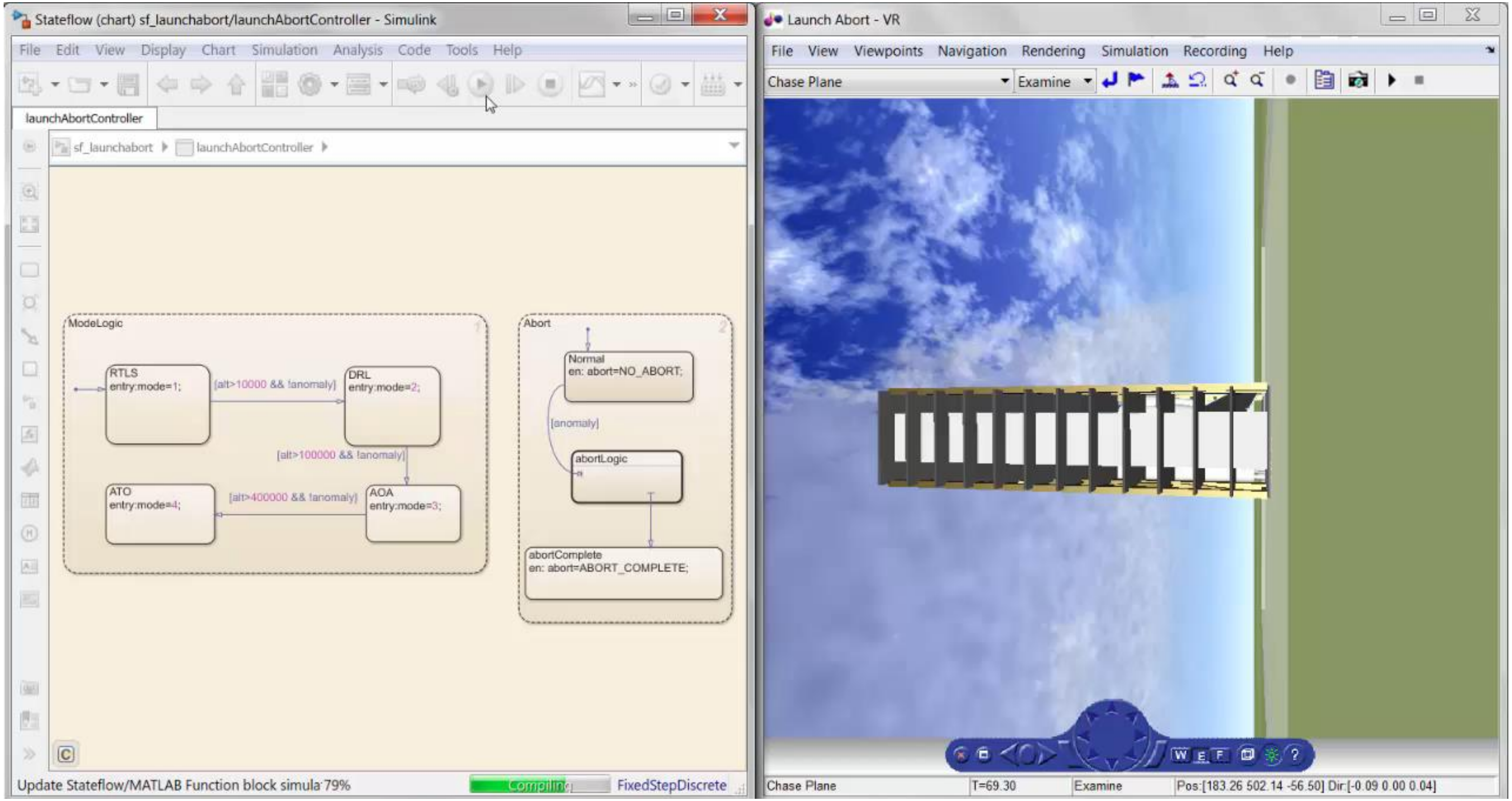
Stateflow

- Model and simulate decision logic
  - supervisory control
  - task scheduling
  - fault management
- Develop mode-logic
  - using state machines and flow charts
- See how the logic behaves with diagram animation
  - and integrated debugger





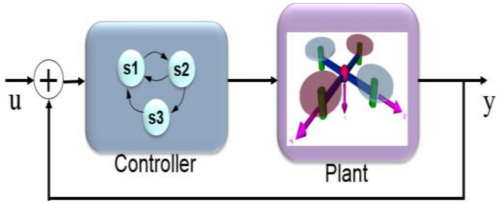
Design **Control algorithm** with Dynamics



Autonomous System Development Workflow

Aerodynamics and Flight Control → **Autonomous algorithm** → **Test and Refine in Simulation** → **Test and Refine on Real Robot**

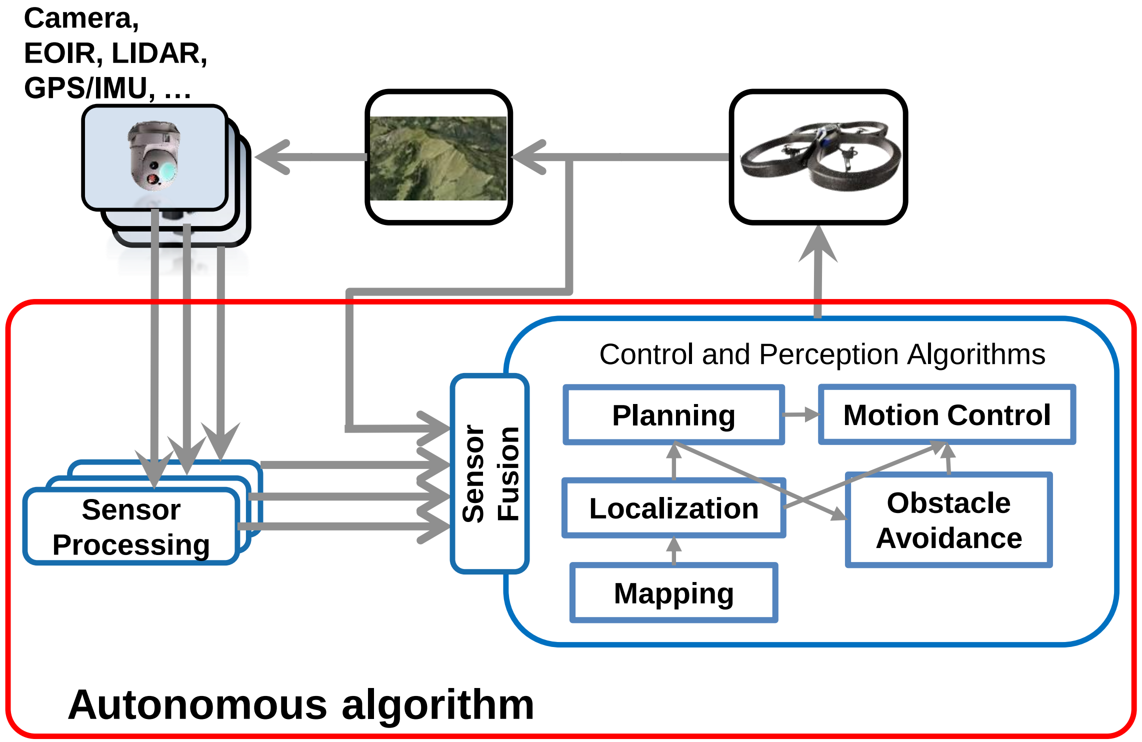
Challenge 1:
Understand
the dynamics
with control algorithm



Challenge 2:
Design and Verify
autonomous
algorithms

Challenge 3:
Verify and Implement
the algorithm
on to a real hardware

How to develop **Autonomous algorithms**?

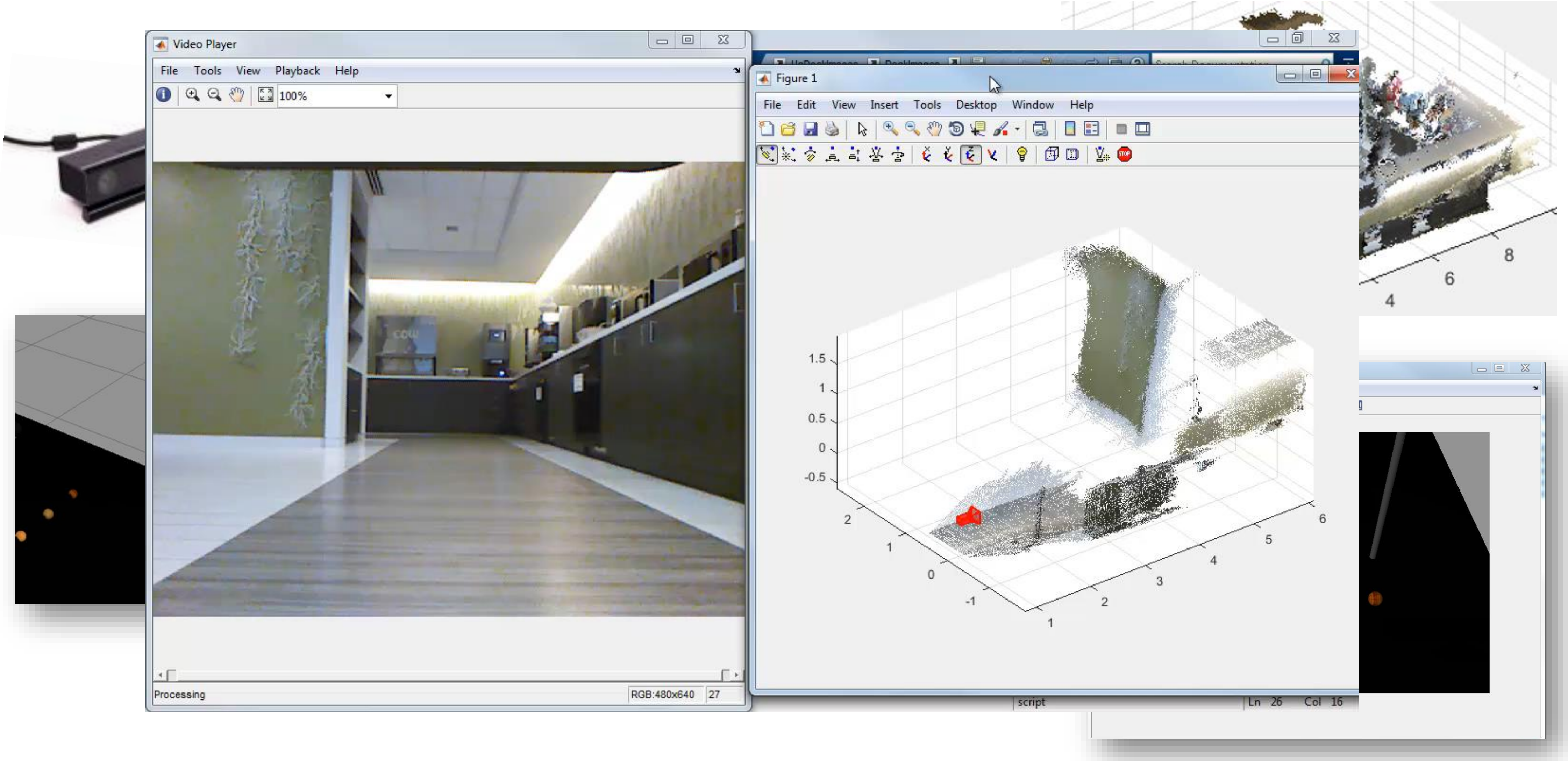


Autoroute, gated, post-process automatically

Efficient, speed, endurance, automation

How to develop **Autonomous algorithms**?

Manage Sensor data



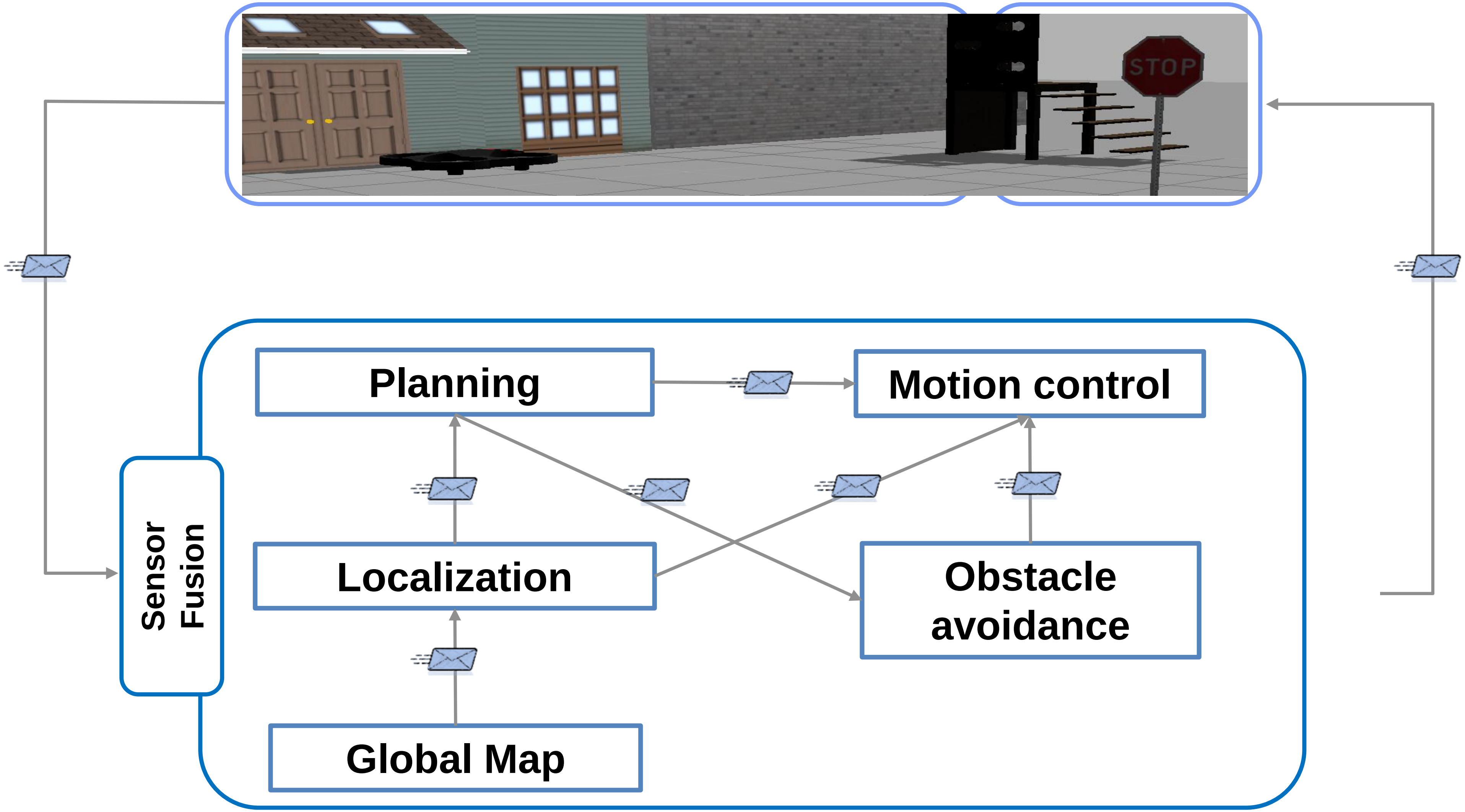
How to develop **Autonomous algorithms**?

Manage Sensor data

Data & RF | Embedded | Imaging | Specialty | Standards

How to develop **Autonomous algorithms**?

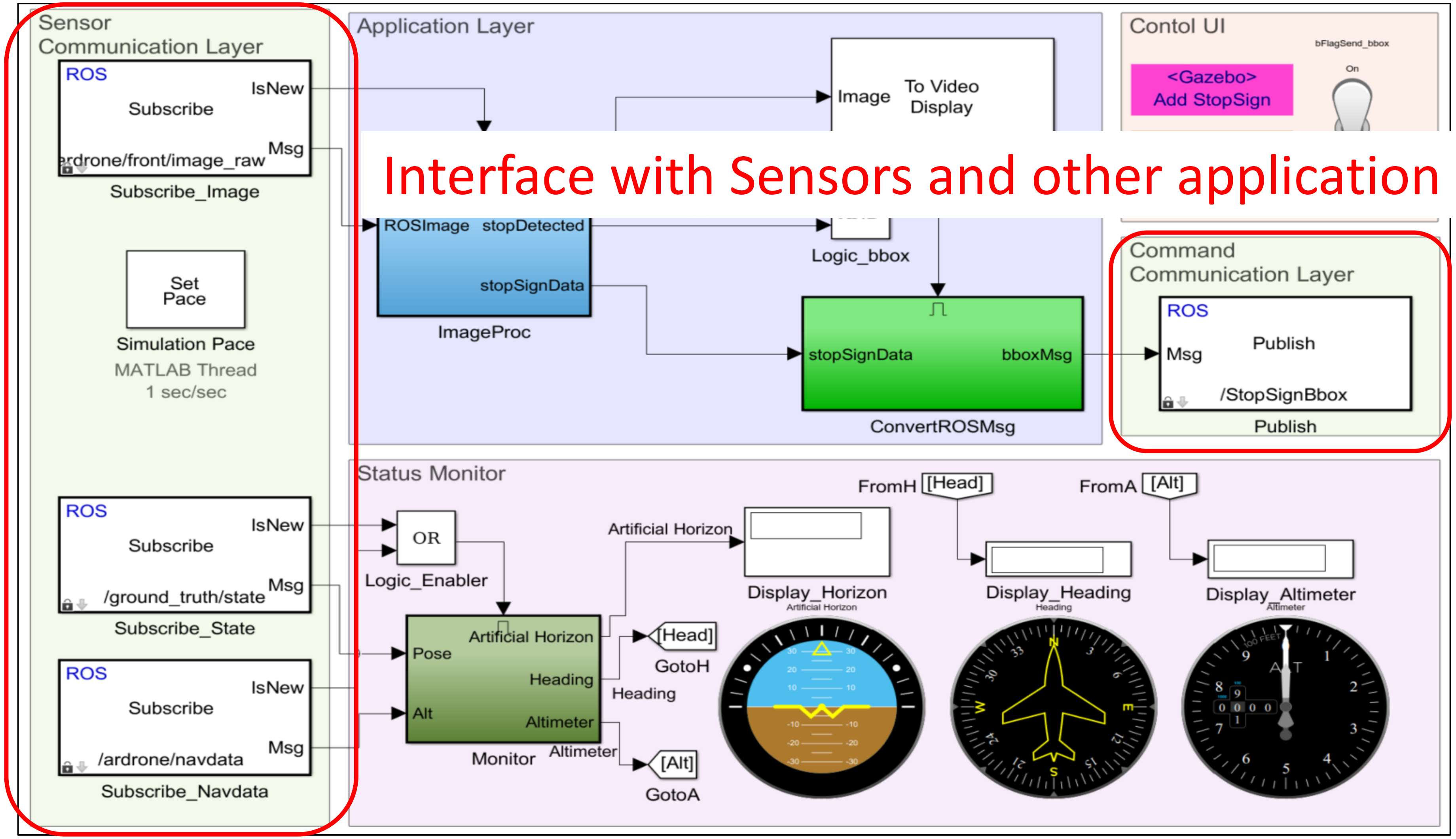
Sensor data flow



EOIR: Electro Optical Infra-Red

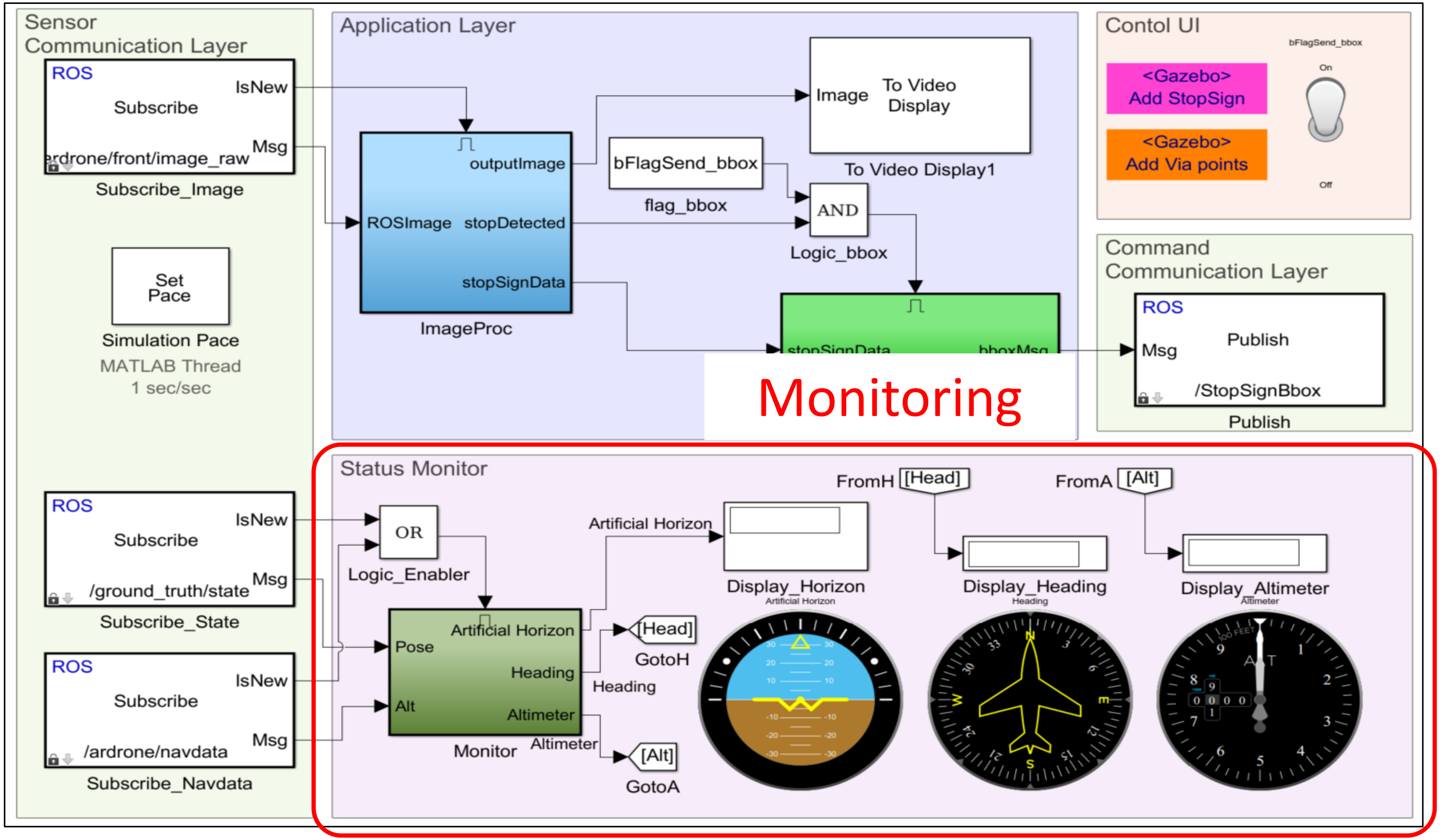
How to develop **Autonomous algorithms**?

System Level Design with MATLAB, Simulink



How to develop **Autonomous algorithms**?

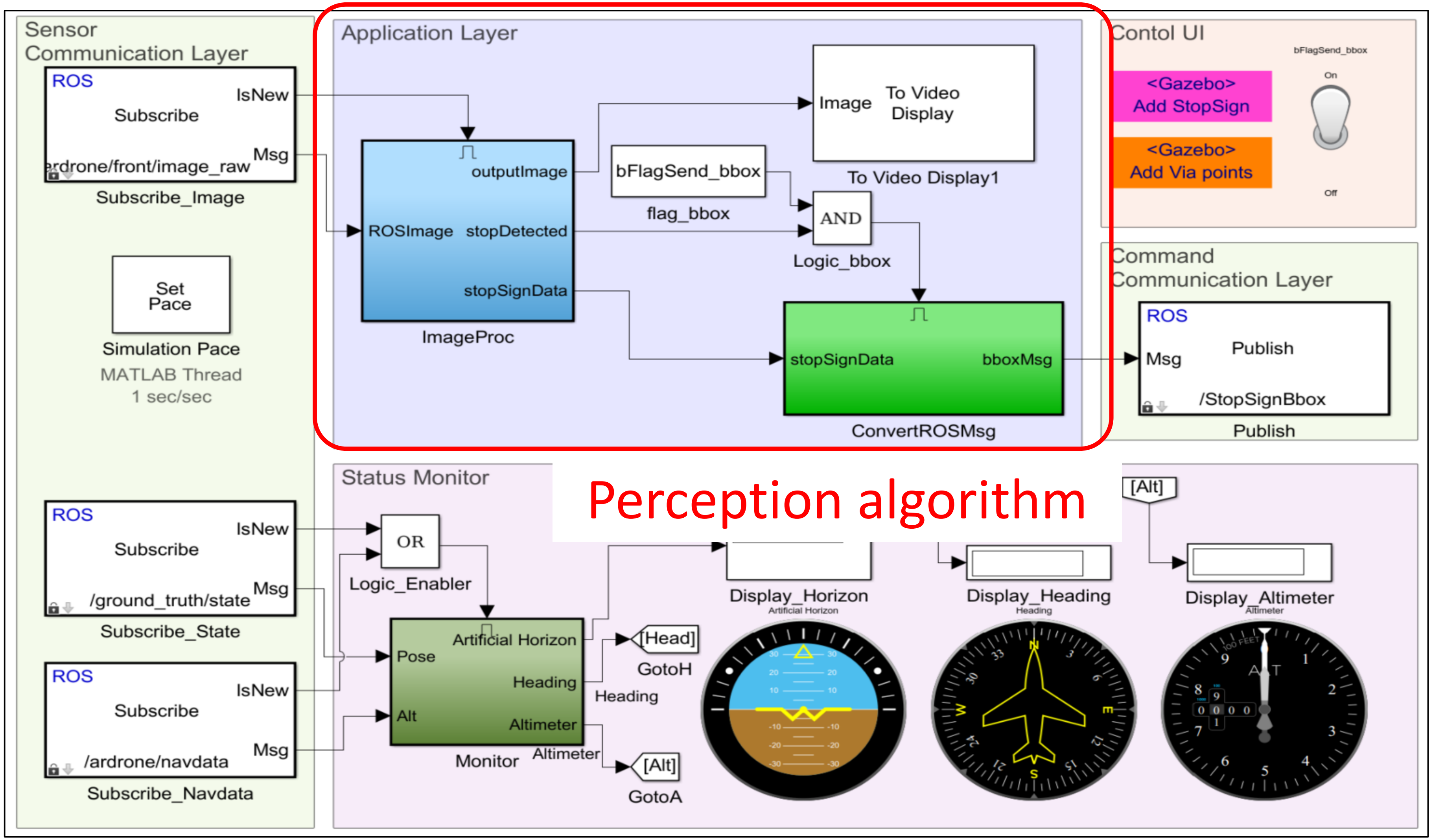
System Level Design with MATLAB, Simulink



Monitoring

How to develop **Autonomous algorithms**?

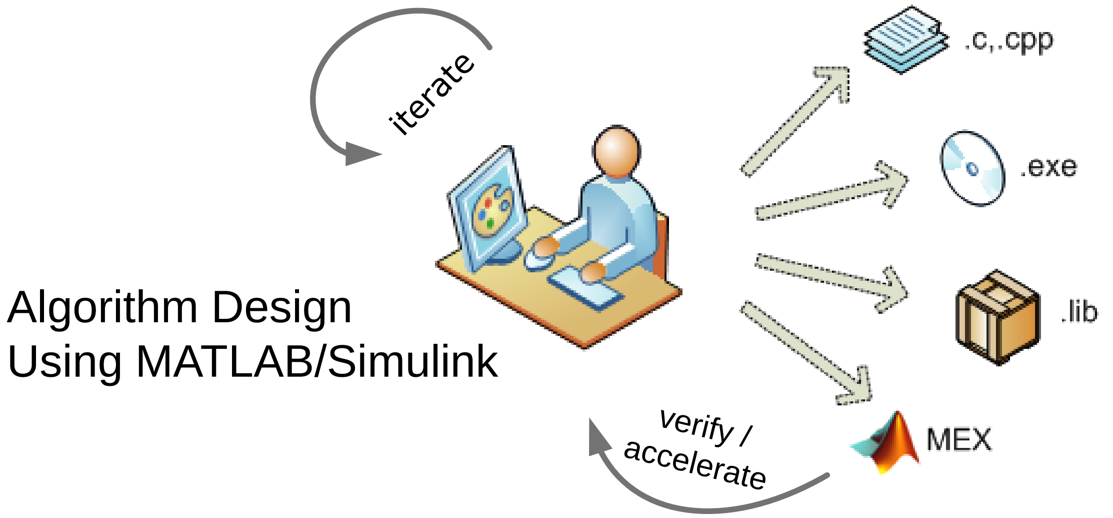
System Level Design with MATLAB, Simulink



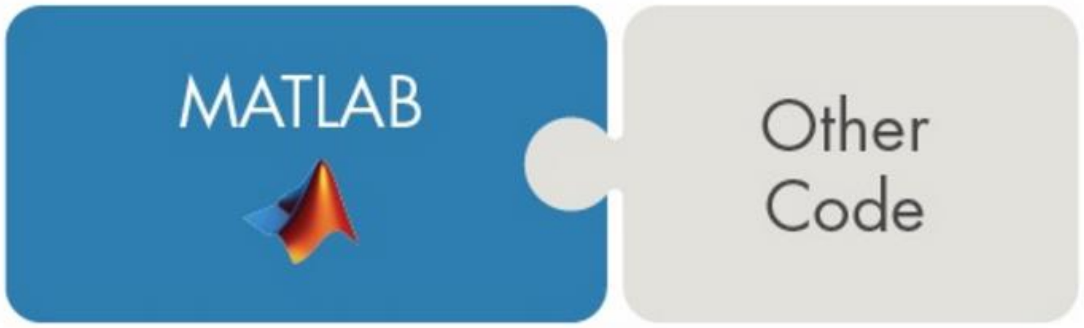
How to develop **Autonomous algorithms?**

System Level Design with MATLAB, Simulink

- Design Perception algorithm
  - Matlab function
  - S-function
  - System Object
  - State & flow control



- Calling Libraries Written in Another Language From MATLAB



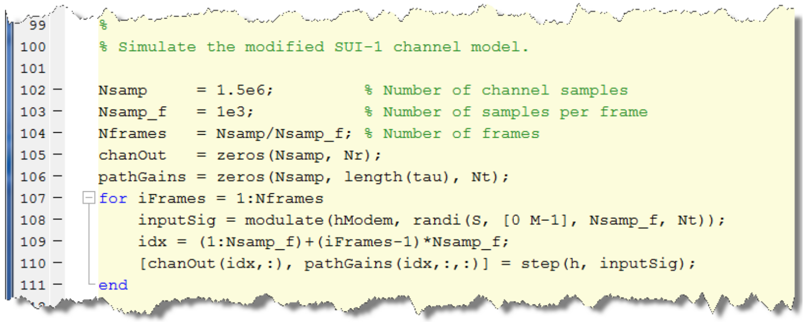
- C/C++, Python, Java
- Fortran
- COM components and ActiveX® controls
- RESTful, HTTP, and WSDL web services

How to develop **Autonomous algorithms**?

System objects

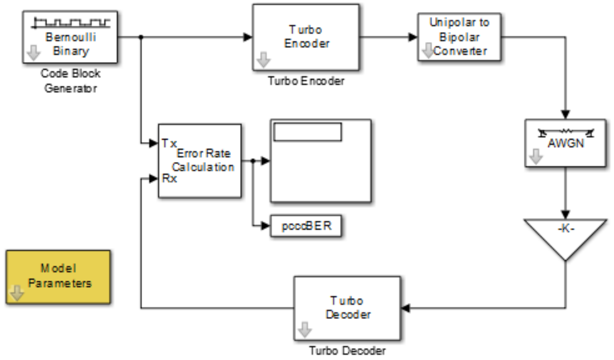
- Model dynamic systems using the MATLAB language
  - Algorithms implemented as MATLAB classes
  - API designed specifically for s ulation
  - Can be used in MATLAB and S

System objects



```
%
% Simulate the modified SUI-1 channel model.

Nsamp     = 1.5e6;          % Number of channel samples
Nsamp_f   = 1e3;            % Number of samples per frame
Nframes   = Nsamp/Nsamp_f;  % Number of frames
chanOut   = zeros(Nsamp, Nr);
pathGains = zeros(Nsamp, length(tau), Nt);
for iFrames = 1:Nframes
    inputSig = modulate(hModem, randi(S, [0 M-1], Nsamp_f, Nt));
    idx = (1:Nsamp_f)+(iFrames-1)*Nsamp_f;
    [chanOut(idx,:), pathGains(idx,:,:)] = step(h, inputSig);
end
```



How to develop **Autonomous algorithms**?

System objects

RemoveMean.m

```
classdef RemoveMean < matlab.System
%RemoveMean Remove estimated running mean

    properties
        % Memory weighting
        Weight = 0.999
    end

    properties (DiscreteState)
        Mean
    end

    methods (Access=protected)
        function y = stepImpl(obj,u)
            y = u - obj.Mean;
            obj.Mean = u - y*obj.Weight;
        end
        function setupImpl(obj)
            obj.Mean = 0;
        end
        function resetImpl(obj)
            obj.Mean = 0;
        end
    end

end
```

Base Class is: matlab.System

Properties provide persistent memory within the object

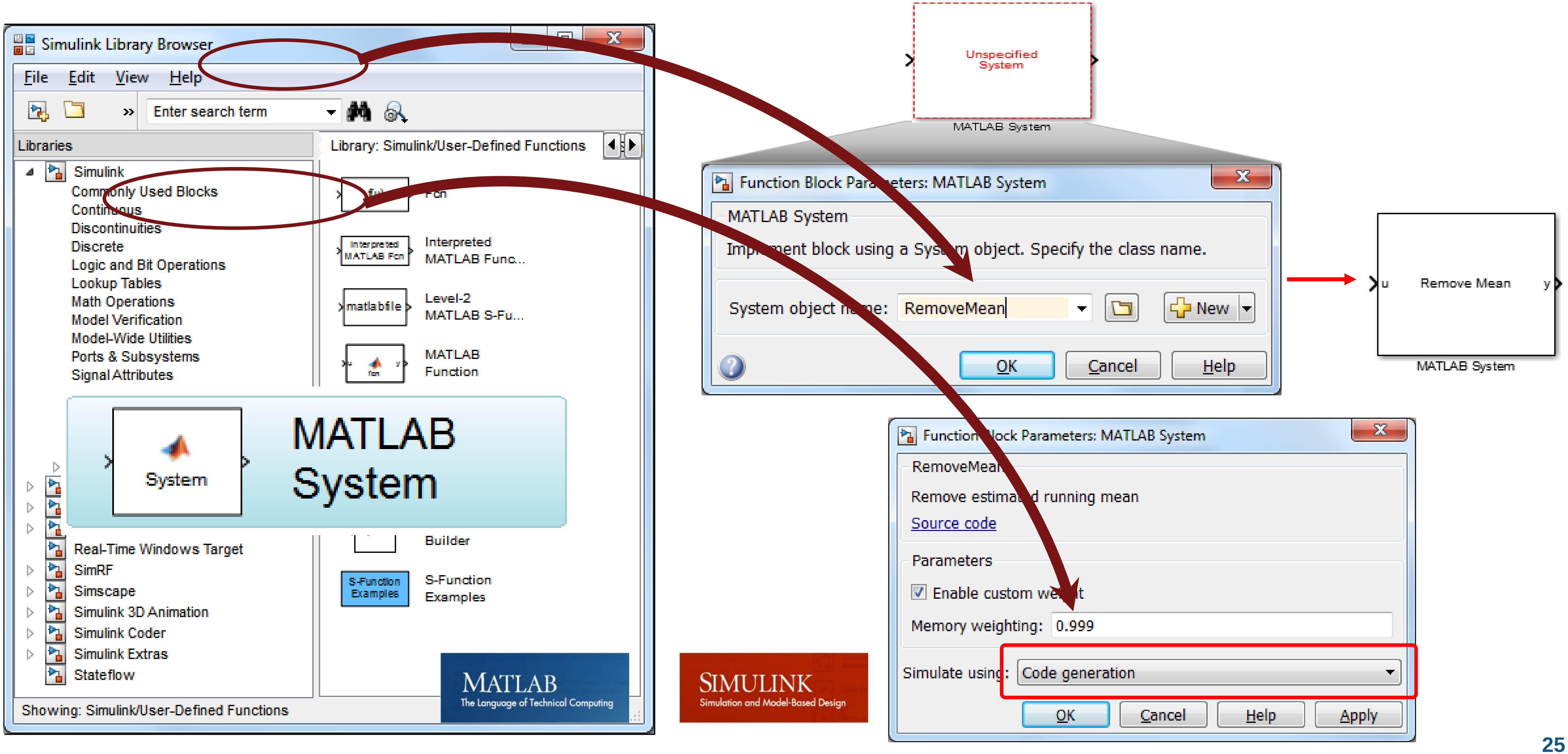
Functions for each phase of your system (validation, initialization, reset, step, ...)

- MATLAB automatically calls housekeeping functions,

e.g., validation, initialization, etc.

How to develop **Autonomous algorithms**?

System objects

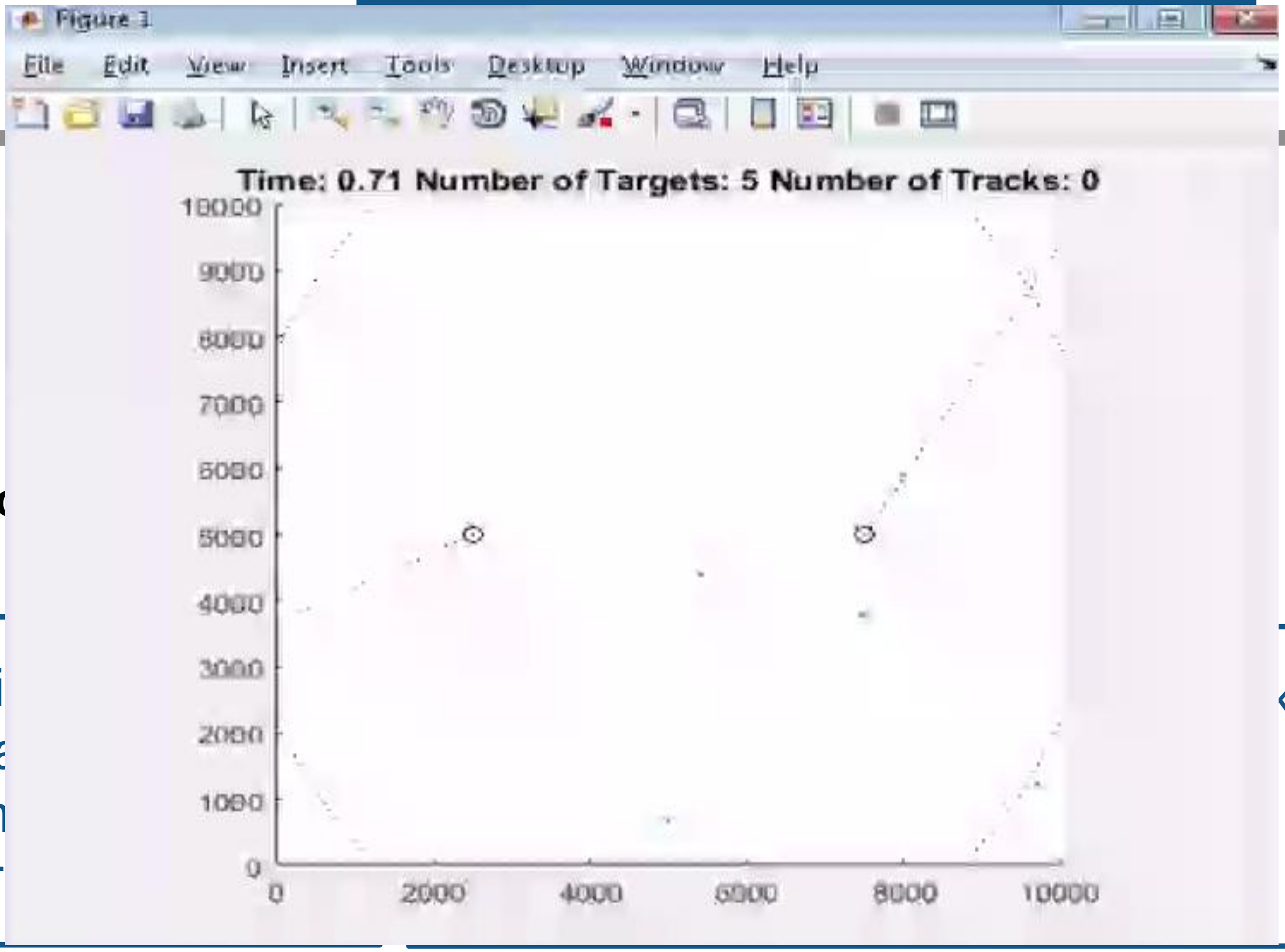


How to develop **Autonomous algorithms?**

Sensor fusion framework

Sensor fusion Framework

Sensors



Tracks

Time
Measurement
Measurement N

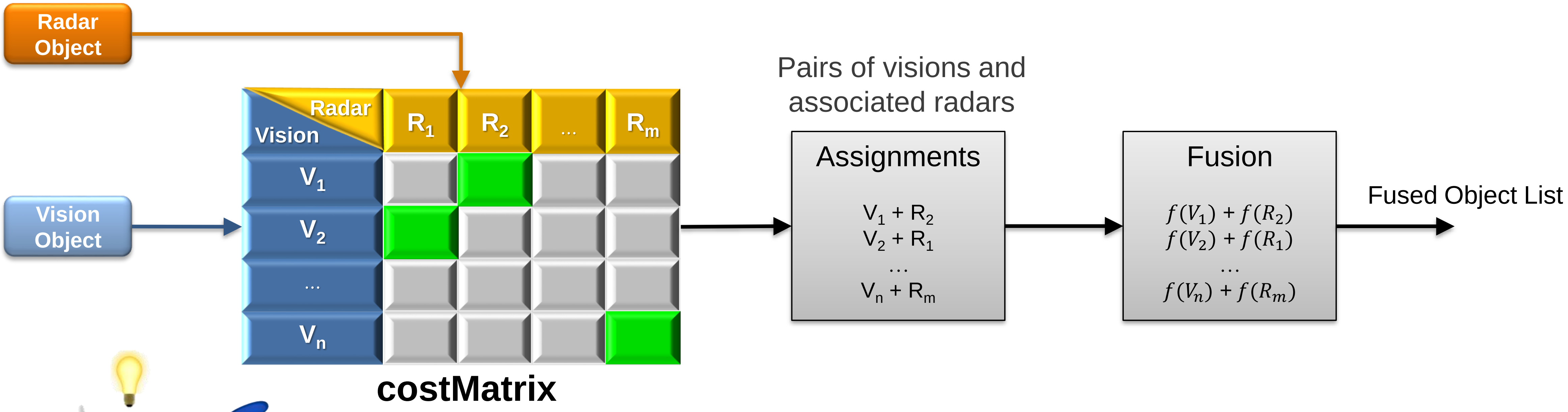
- Assigns detecti
- Creates new tra
- Updates existin
- Removes old tr

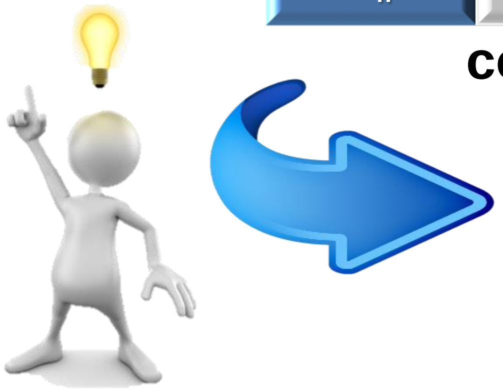
Time
State
State Covariance
Track ID
Age
Is Confirmed
Is Coasted

How to develop **Autonomous algorithms**?

Sensor fusion framework

- Track Manager



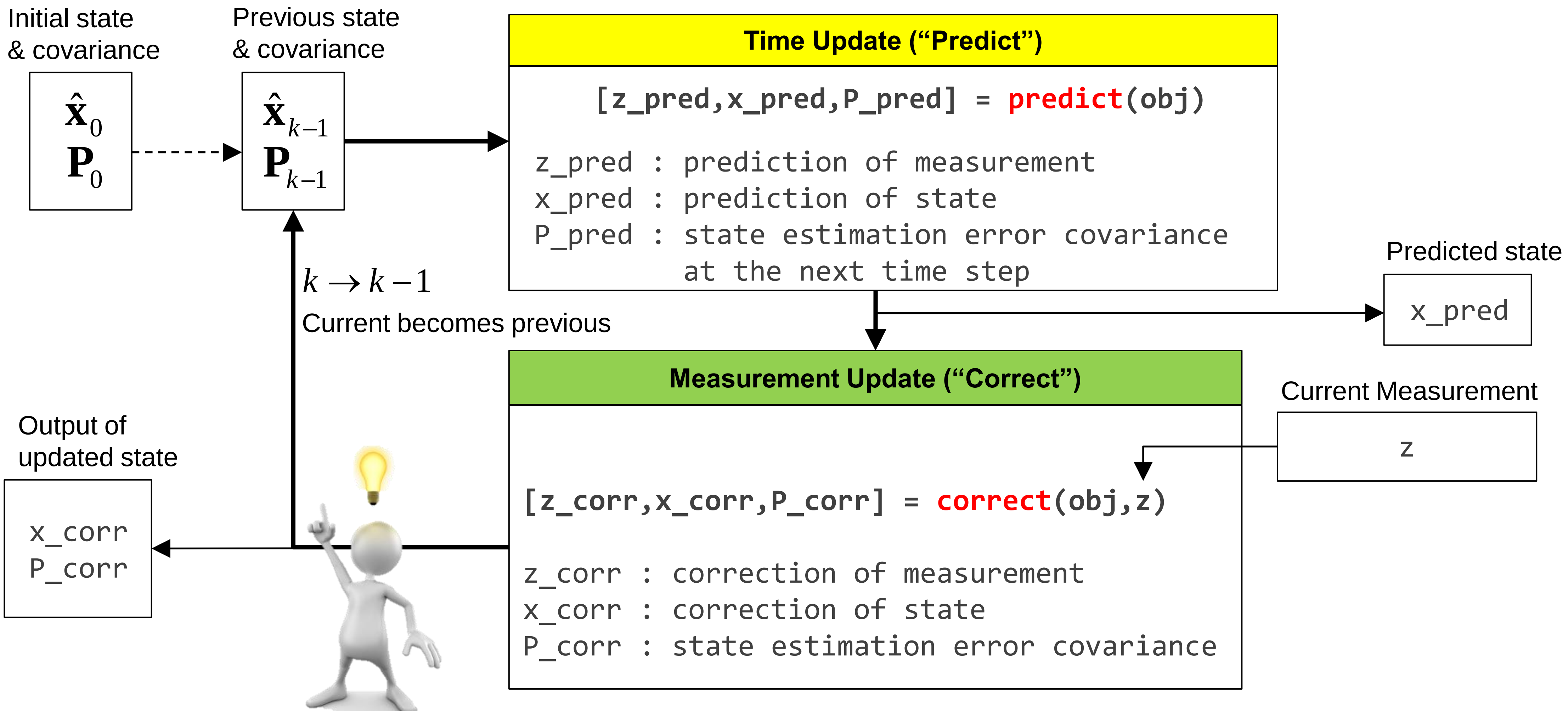


```
[assignments, unassignedVisions, unassignedRadars] = ...
    assignDetectionsToTracks(costMatrix, param.costOfNonAssignment);
```

How to develop **Autonomous algorithms**?

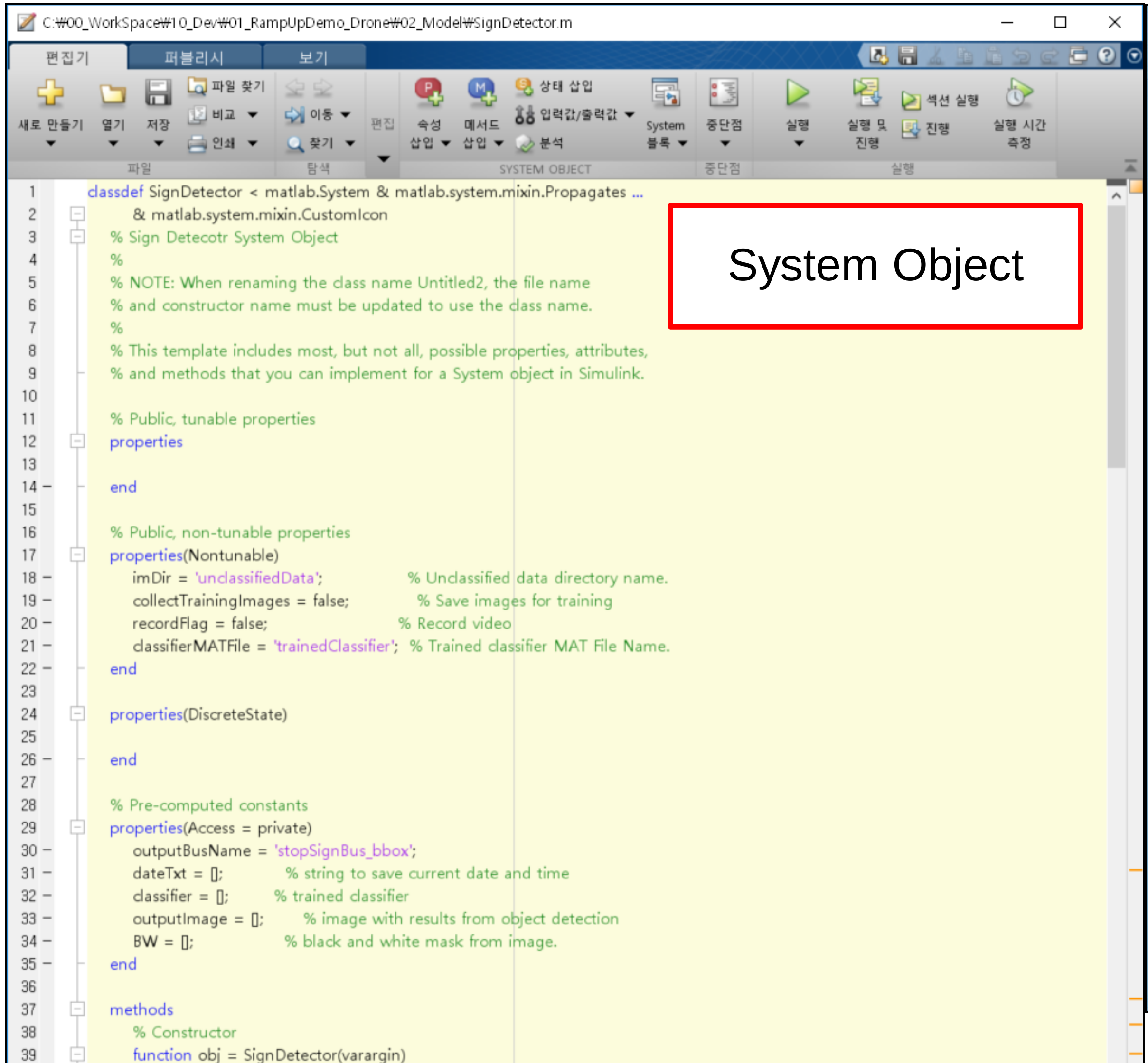
Sensor fusion framework

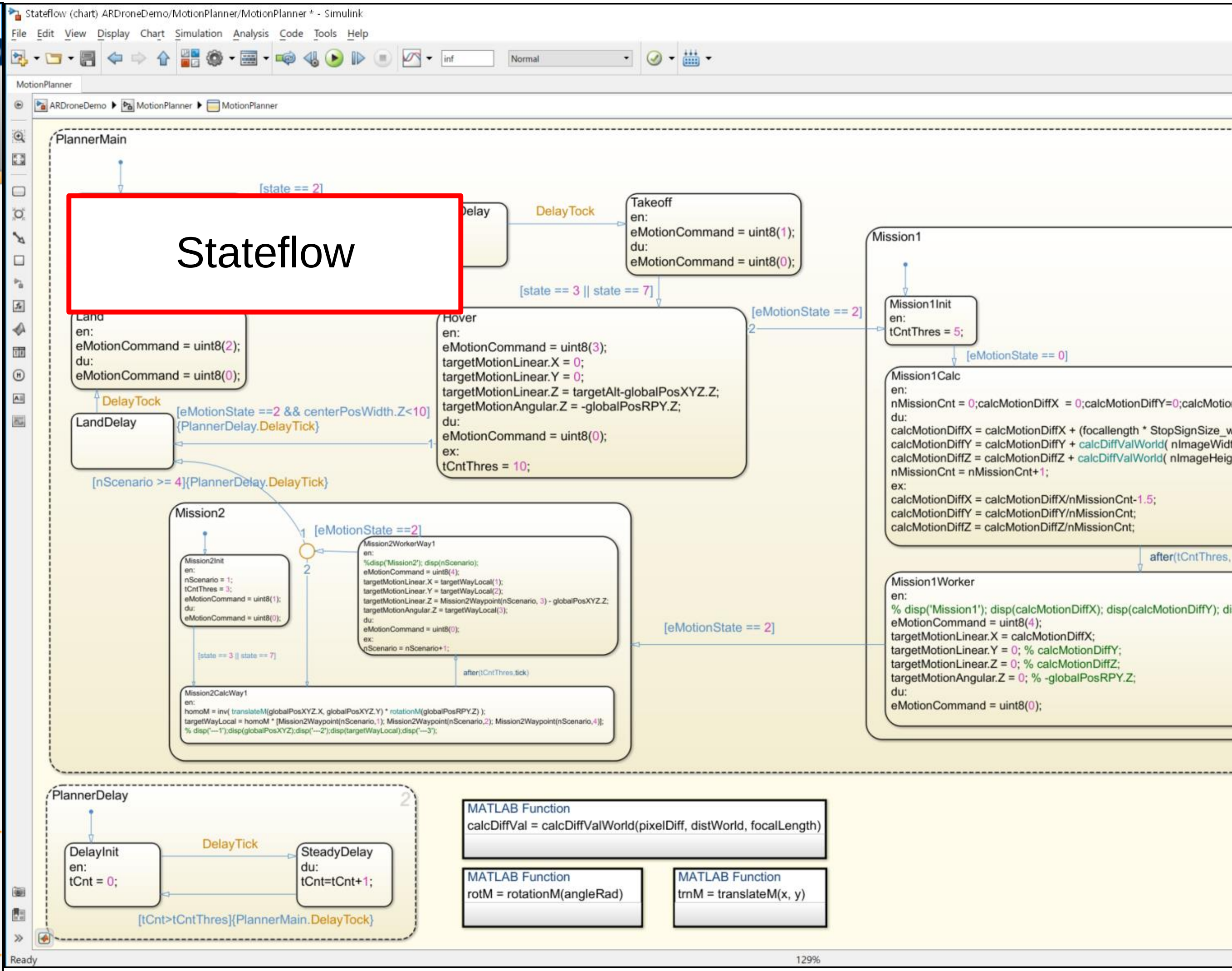
- Tracking Filter



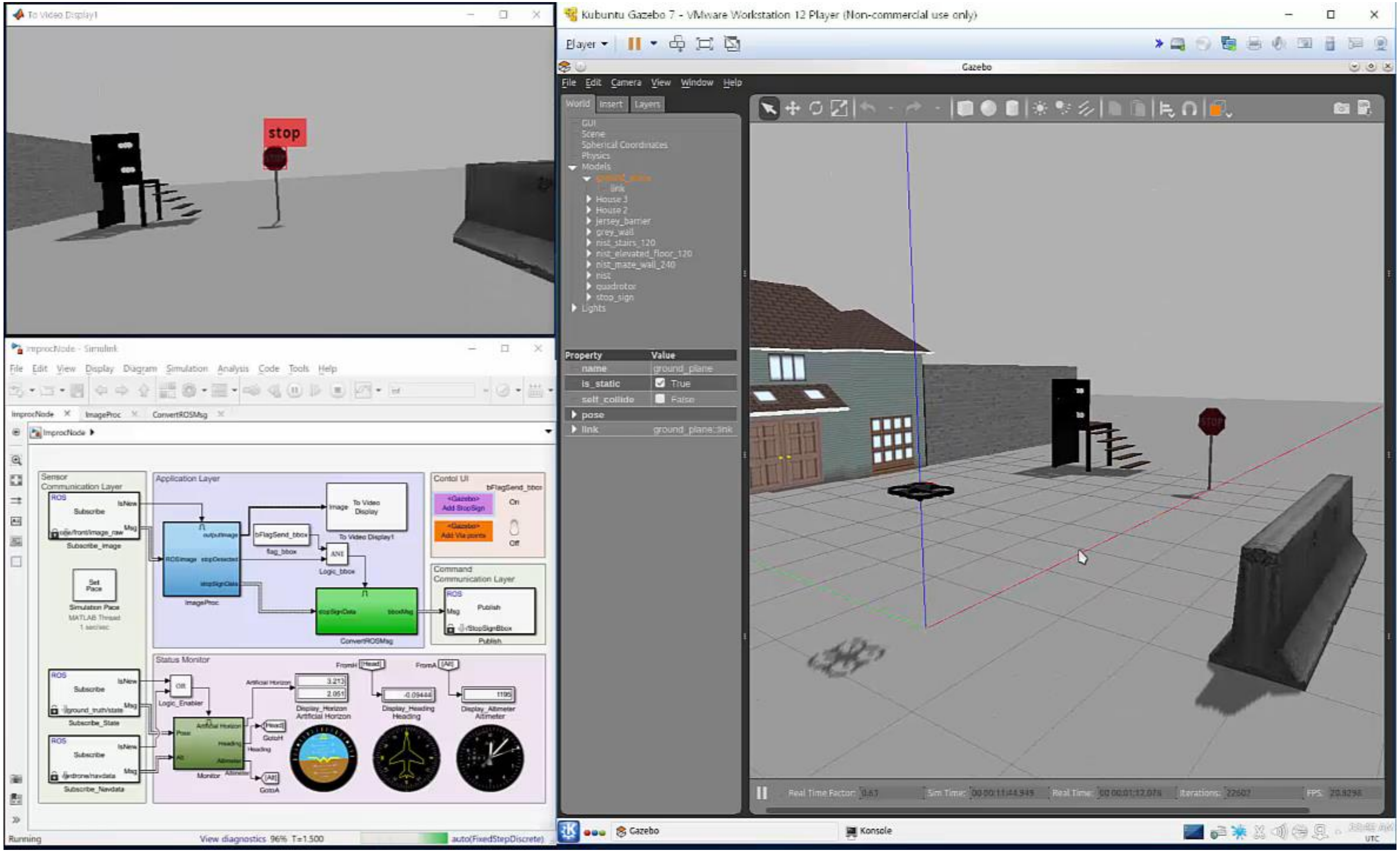
How to develop **Autonomous algorithms**?

System Level Design with MATLAB, Simulink





Design (and verify) **Autonomous algorithms**



How to develop **Autonomous algorithms**?

CONTROL

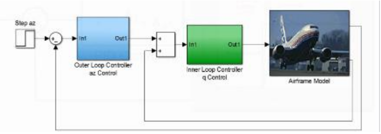
SENSE

PERCEIVE

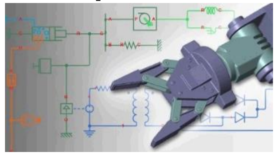
PLAN

CONNECT

Control System Tbx

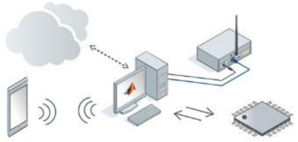


Simscape Toolboxes

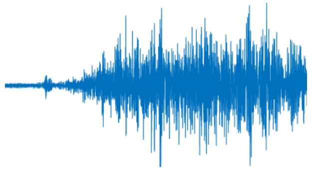


Simulink Real-Time

HW Support Packages

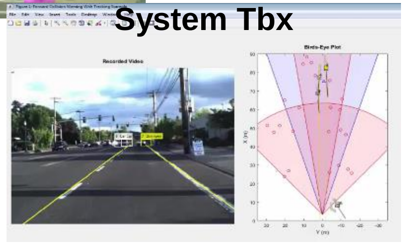


Data Acquisition Tbx

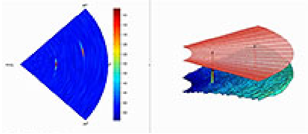


Computer Vision

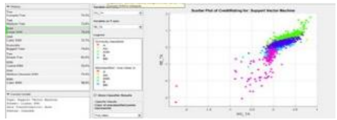
Automated Driving System Tbx



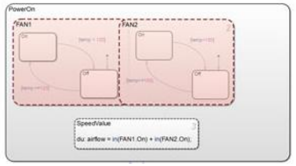
Phased Array



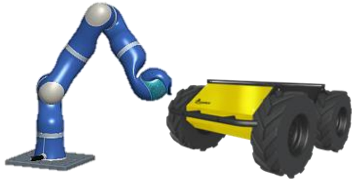
Statistics & Machine Learning



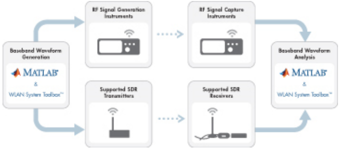
Stateflow



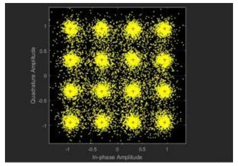
Robotics System Tbx



Communications Tbx



WLAN System Toolbox



Robotics System Tbx

ROS

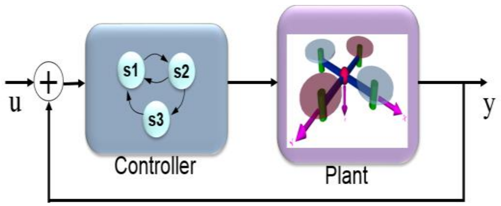
Autonomous System Development Workflow

Aerodynamics and Flight Control → **Autonomous algorithm** → **Test and Refine in Simulation** → **Test and Refine on Real Robot**

Challenge 1:
Understand
the dynamics
with control algorithm

Challenge 2:
Design and Verify
autonomous
algorithms

Challenge 3:
Verify and Implement
the algorithm
on to a real hardware



How to **Deploy** autonomous algorithm?

MATLAB Coder - Code from MATLAB

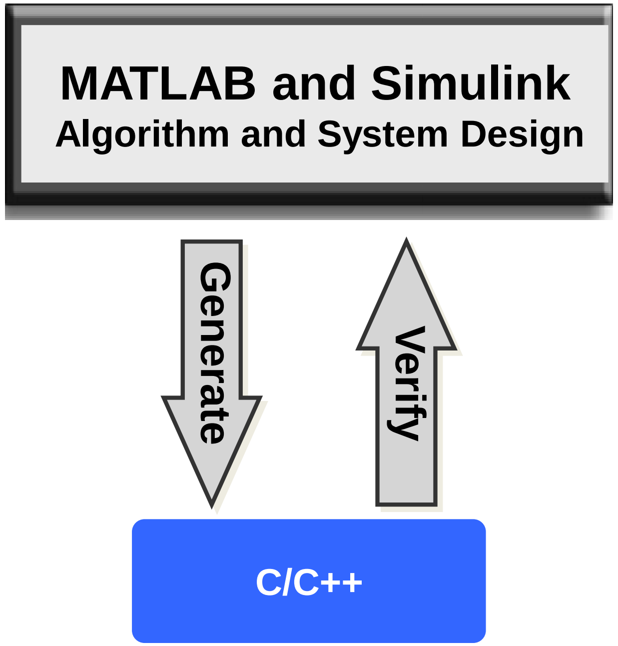
- Portable code for numerical algorithms
- Desktop applications (standalone, library)

Simulink Coder - Code from Simulink

- Rapid prototyping or HIL applications
- Real-time machines

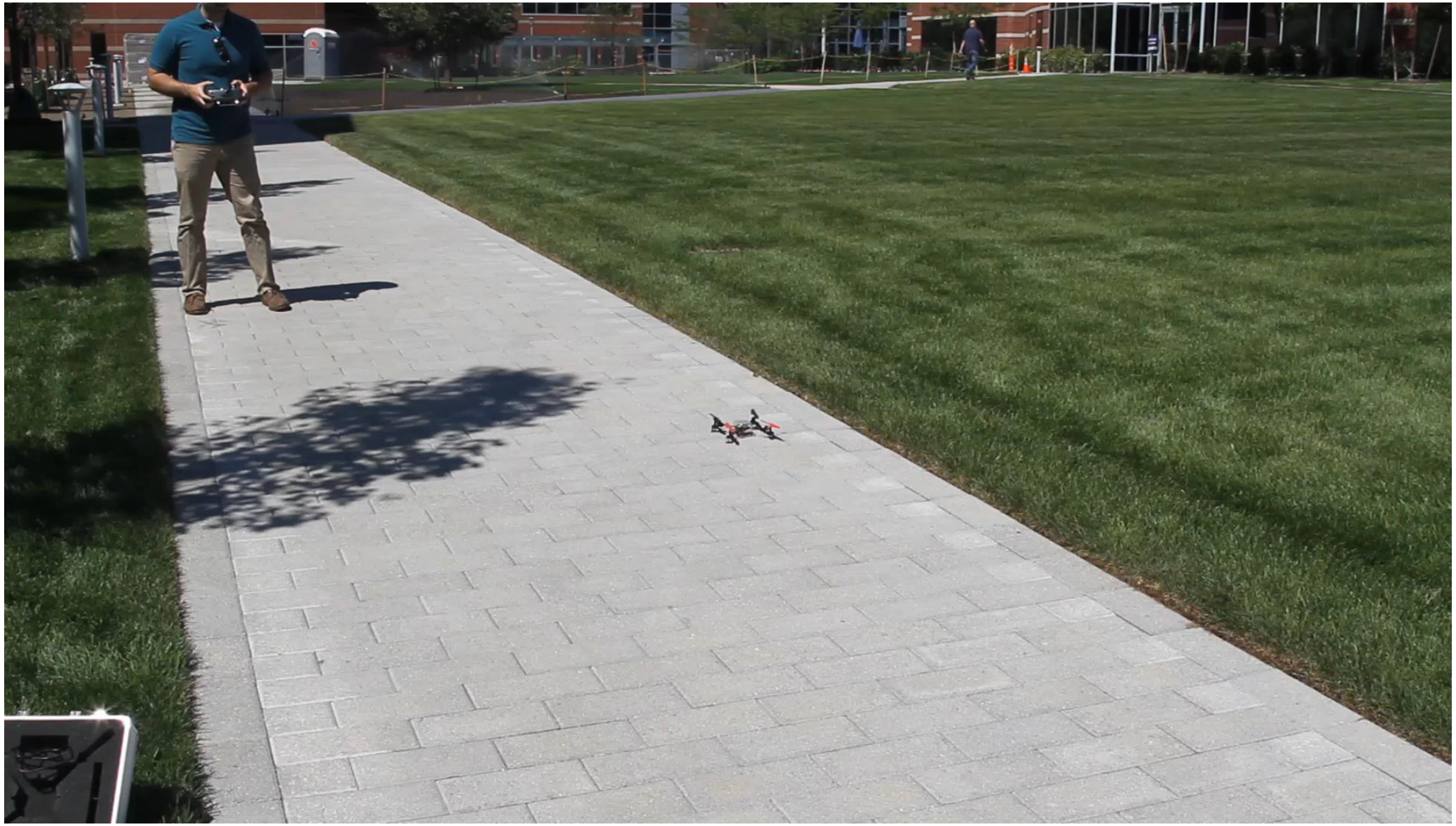
Embedded Coder – Production optimized code

- Embedded applications
- MCU and DSP (fixed or float)
- Code verification (in-the-loop)
- Target-specific support (APIs and examples)



All coders generate portable code (ANSI/ISO C) by default.

How to **Deploy** autonomous algorithm?



Autonomous System with MATLAB/Simulink

BAE Systems Controls Develops Autopilot for Unmanned Aerial Vehicle Using MathWorks Tools

An Eagle 150 unmanned aerial vehicle flight.

(Image courtesy of Composites Technology Research Malaysia.)

Challenge

Enable teams working in separate locations to design a sophisticated UAV autopilot system quickly and inexpensively

Solution

Use MathWorks tools, modify existing software designs with Model-Based Design, and automatically generate embedded control code

Results

- Design and rework costs substantially reduced
- Testing cycle time minimized
- Coding errors and manual documentation work minimized

"MATLAB and Simulink greatly reduced development cycle time and cut system software design and testing costs by 50%."

Feng Liang
BAE Systems Controls

Link to user story

Autonomous System with MATLAB/Simulink

Airnamics Develops Unmanned Aerial System for Close-Range Filming with Model-Based Design

Airnamics co-founders Marko Thaler and Zoran Bjelić with the R5 MSN1 prototype after its first flight.

Challenge

Design and develop an unmanned aerial camera motion system for close-range aerial filming

Solution

Use Model-Based Design with MATLAB and Simulink to accelerate the design, debugging, and implementation of the vehicle's fly-by-wire and flight management system software

Results

- Time-to-market shortened by up to an order of magnitude
- Test flight anomalies quickly resolved
- Debugging time reduced from weeks to hours

"With Model-Based Design our three-engineer team found more than 95% of control software bugs before the first flight. We used the test flights to increase our Simulink models' fidelity and isolate remaining bugs with high precision. The result is a safer, more reliable, and higher-quality product."

Marko Thaler

Airnamics

Link to user story

Summary of Aerial Autonomous System

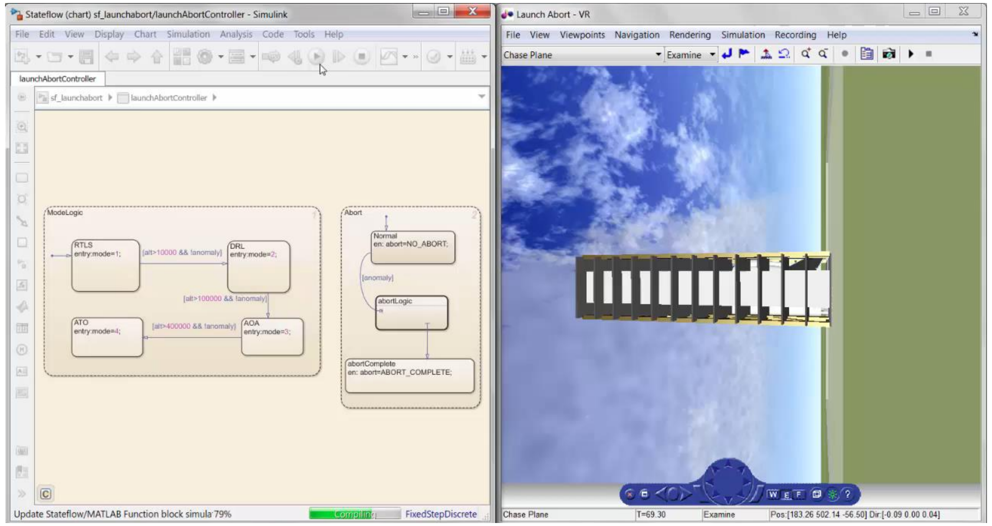
"Autonomous Algorithms"

Aircraft	Perception		Planning		Communication
Platform	**Sense**	**Perceive**	**Plan & Decide**	**Control**	**Connect/ Communicate**
• Control Surfaces, slats, flaps • Lifting Body • Landing Gear • Battery • Power Management	• Radar • Camera • Lidar • EO/IR • IMU • GPS-INS • HW Certification	• Environment mapping • Classification • Segmentation • Object Detection • Sensor Fusion	• Object Avoidance • Path & motion planning • SLAM	• Guidance, Navigation & Control • Flight SW certification	• Communication with ground operator • Multi-agent communication • Satellite data link

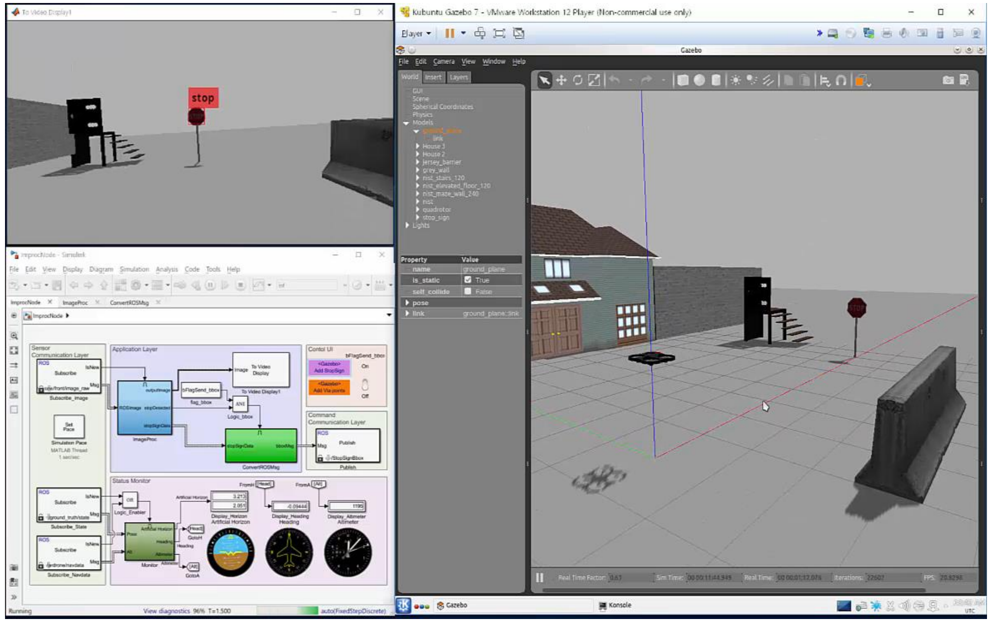
Key Takeaway

Designing Autonomous system
using MATLAB and Simulink can help in :

- **Understanding the dynamics with control algorithm**
  - Model aerodynamics, propulsion and motion
  - Design control algorithm in single environment



- **Design vision, radar, perception algorithms**
  - Visualizing different sensor data
  - Develop and test sensor fusion and tracking algorithm



- **Implementing the algorithm on actual hardware**
  - Test and verify algorithm on 3D simulators
  - Automatic C/C++ code generation on to actual hardware

% Thank you

MATLAB

is the **easiest** and
most **productive** environment
for **engineers** and **scientists**