MATLAB EXPO 2018

Introduction à Simulink et Stateflow

Olivier Berard

Topics we will address this session

- Why model a system?
- Why use Simulink?
- Getting to grips with the basics of Simulink and Stateflow through a worked example

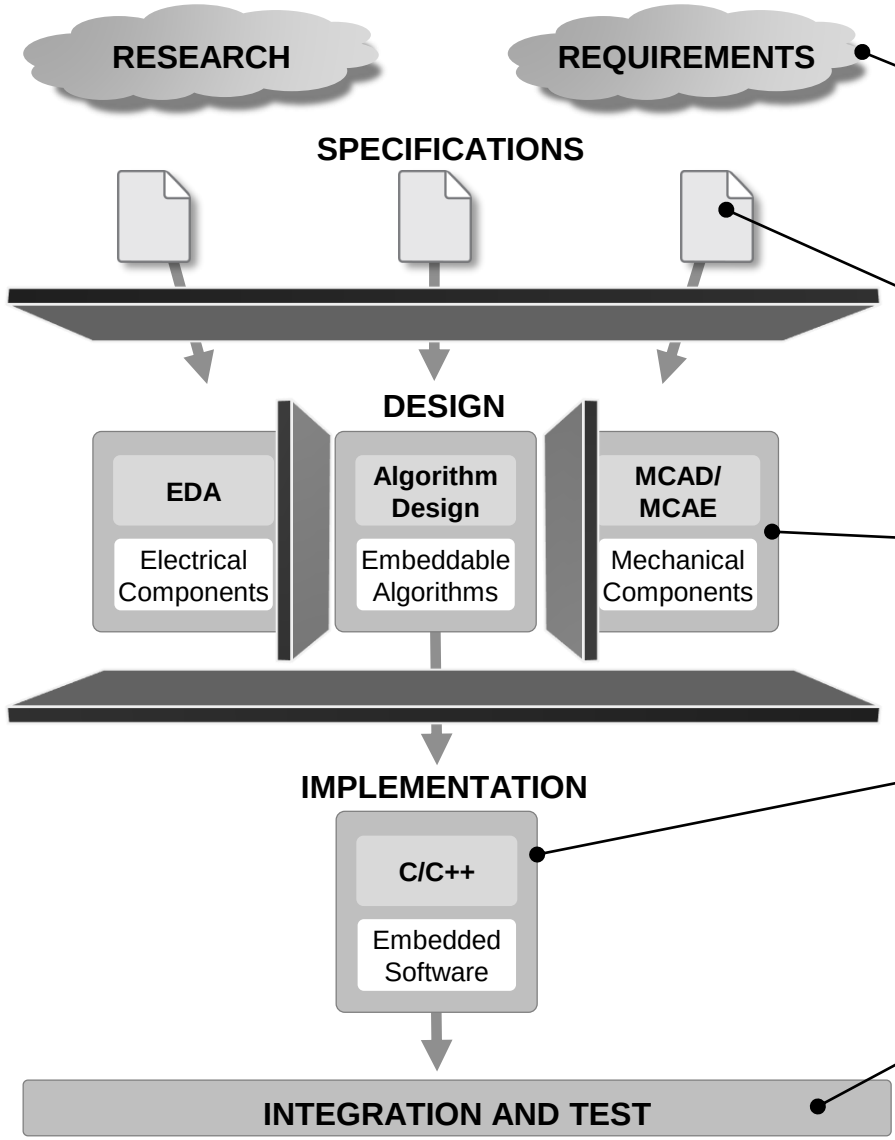
Why model a system?

Image credit: McLaren

Image credit: Peter Gronemann | Wikipedia

Modelling & Simulation gives you insight

Traditional Development Process



Requirement Documents
- Difficult to analyze
- Difficult to manage as they change

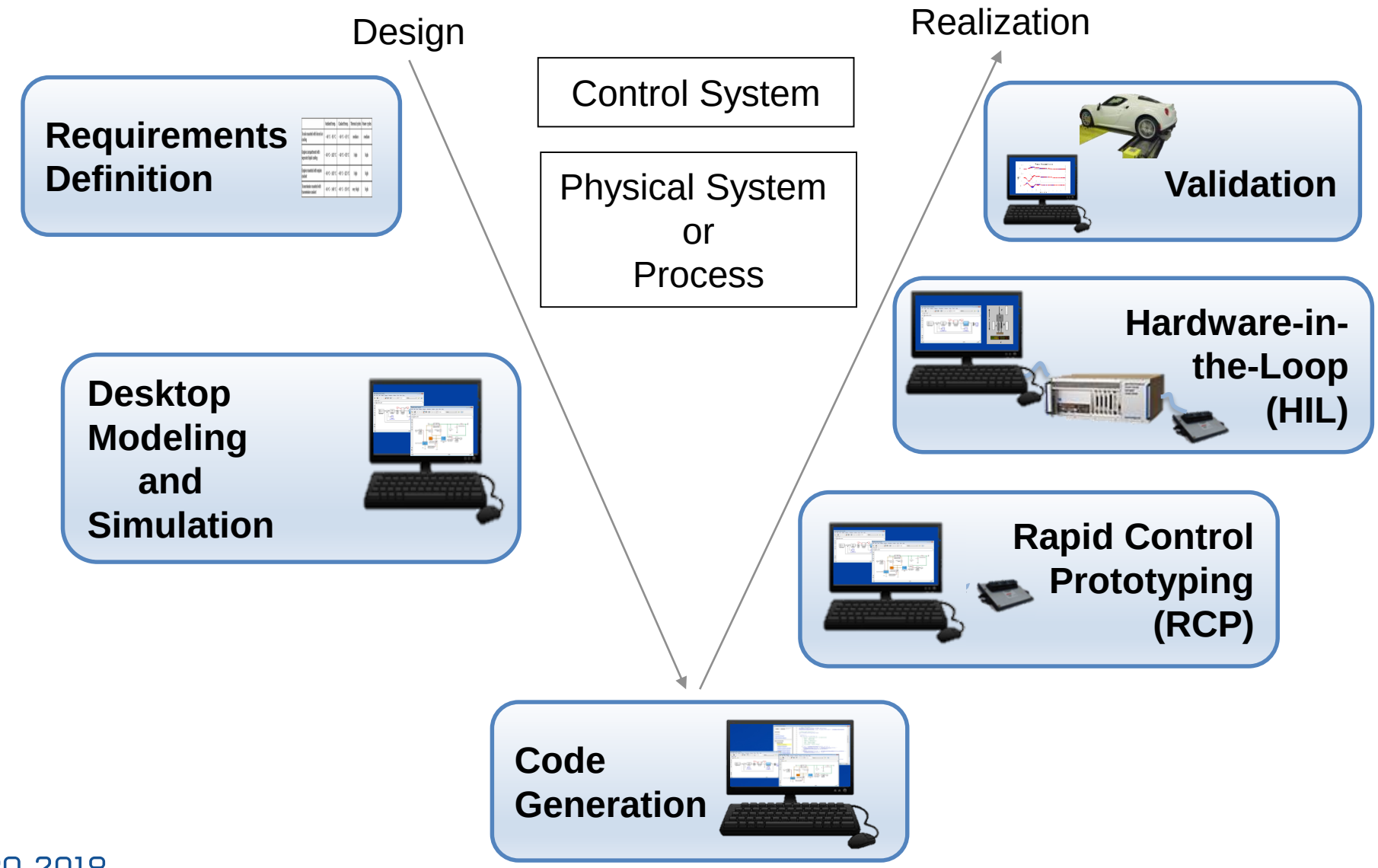
Paper Specifications
- Easy to misinterpret
- Difficult to integrate with design

Physical Prototypes
- Incomplete and expensive
- Prevents rapid iteration
- No system-level testing

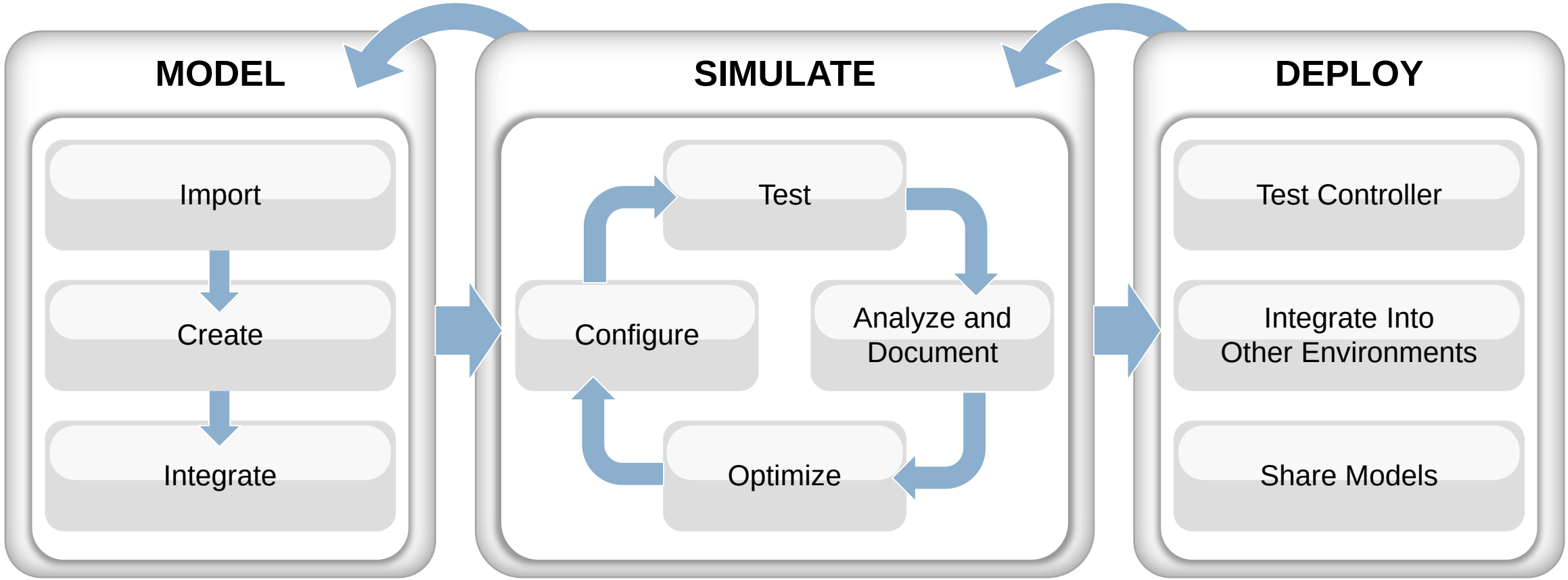
Manual Coding
- Time consuming
- Introduces defects and variance
- Difficult to reuse

Traditional Testing
- Design and integration issues found late
- Difficult to feed insights back into design process
- Traceability

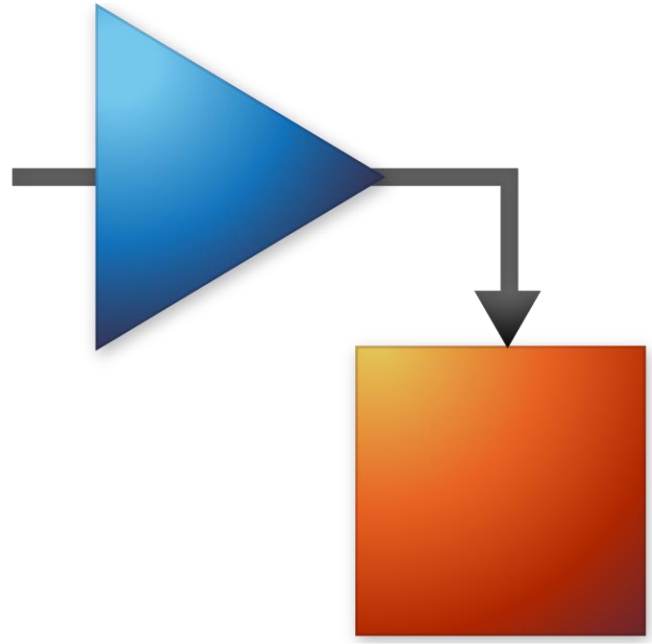
Model-Based Design



Model-Based Design

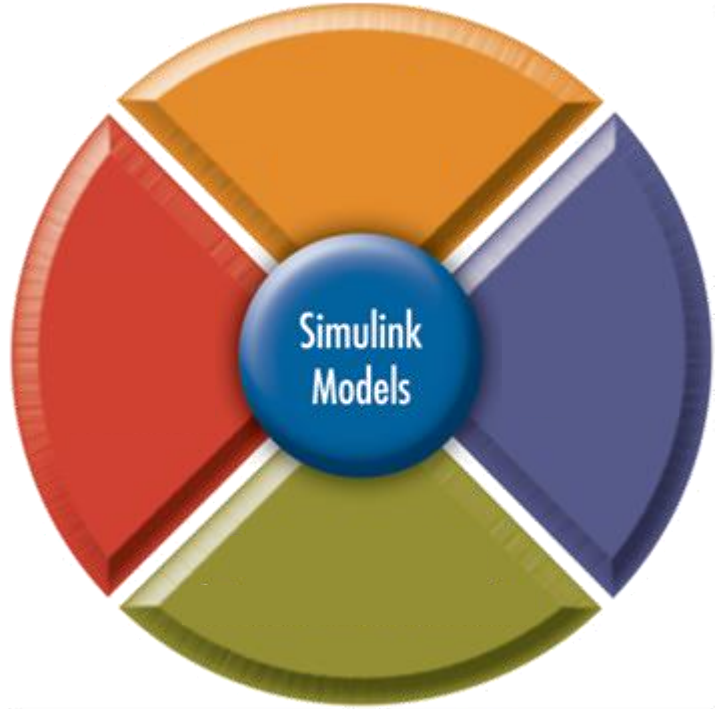


Why use Simulink?

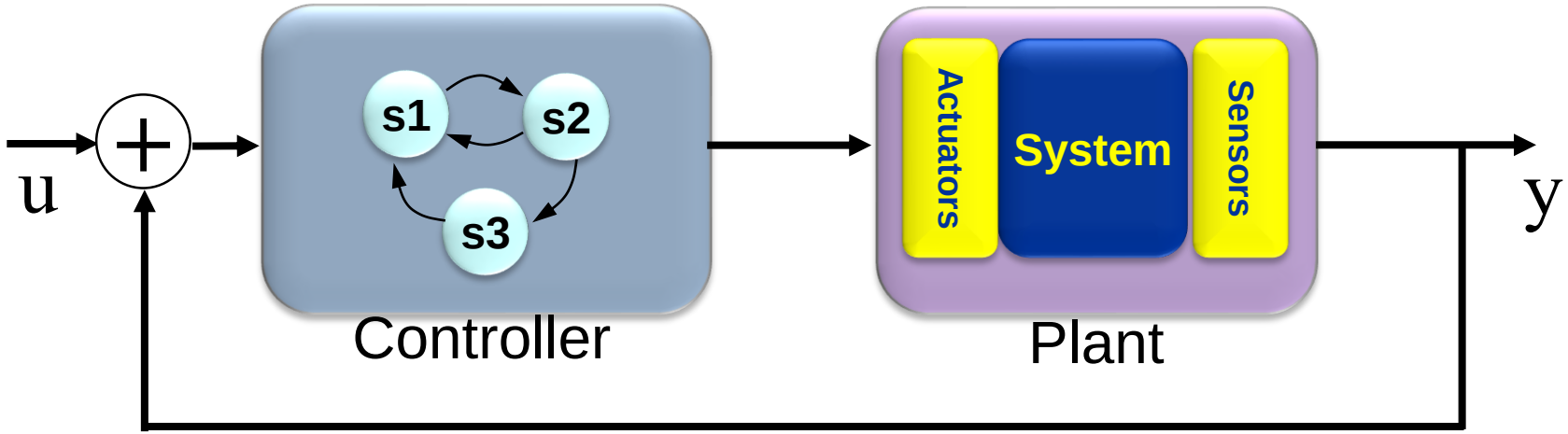


Model Based Design with Simulink

- Modelling and simulation
  - Multidomain Dynamic Systems
  - Nonlinear Systems
  - Continuous-time, Discrete-time, Multi-Rate systems
- Plant and Controller Design
  - Select/optimise control architecture and parameters
  - Rapidly model “what-if” scenarios
  - Communicate design ideas
  - Embody performance specifications
- Implementation
  - Automatic code generation
    - Embedded systems, FPGAs, GPUs
  - Rapid prototyping for HIL, SIL, PIL
  - Verification and validation

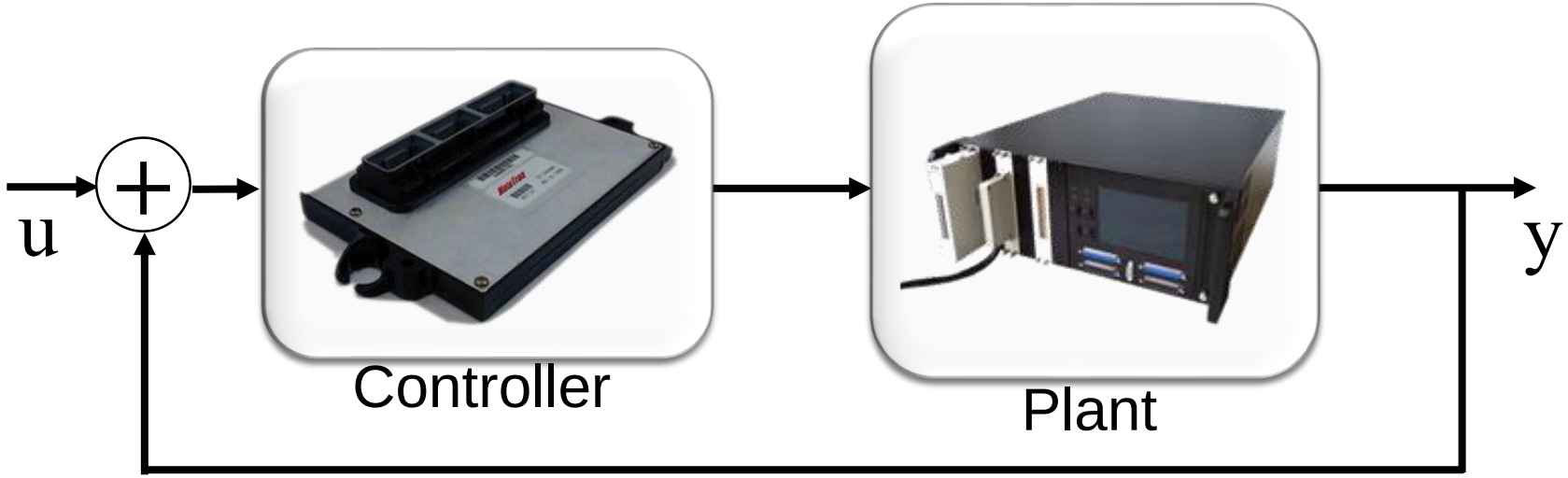


Optimise System-Level Performance



- Simulating plant and controller **in one environment** allows you to **optimize system-level performance**.
  - Automate tuning process using optimization algorithms
  - Accelerate process using parallel computing

Detect Integration Issues Earlier



- Controls engineers and domain specialists can work together to **detect integration issues in simulation**
  - Convert plant models to C code for hardware-in-the-loop tests
  - Share models with other internal users
  - Share models with external users while protecting IP

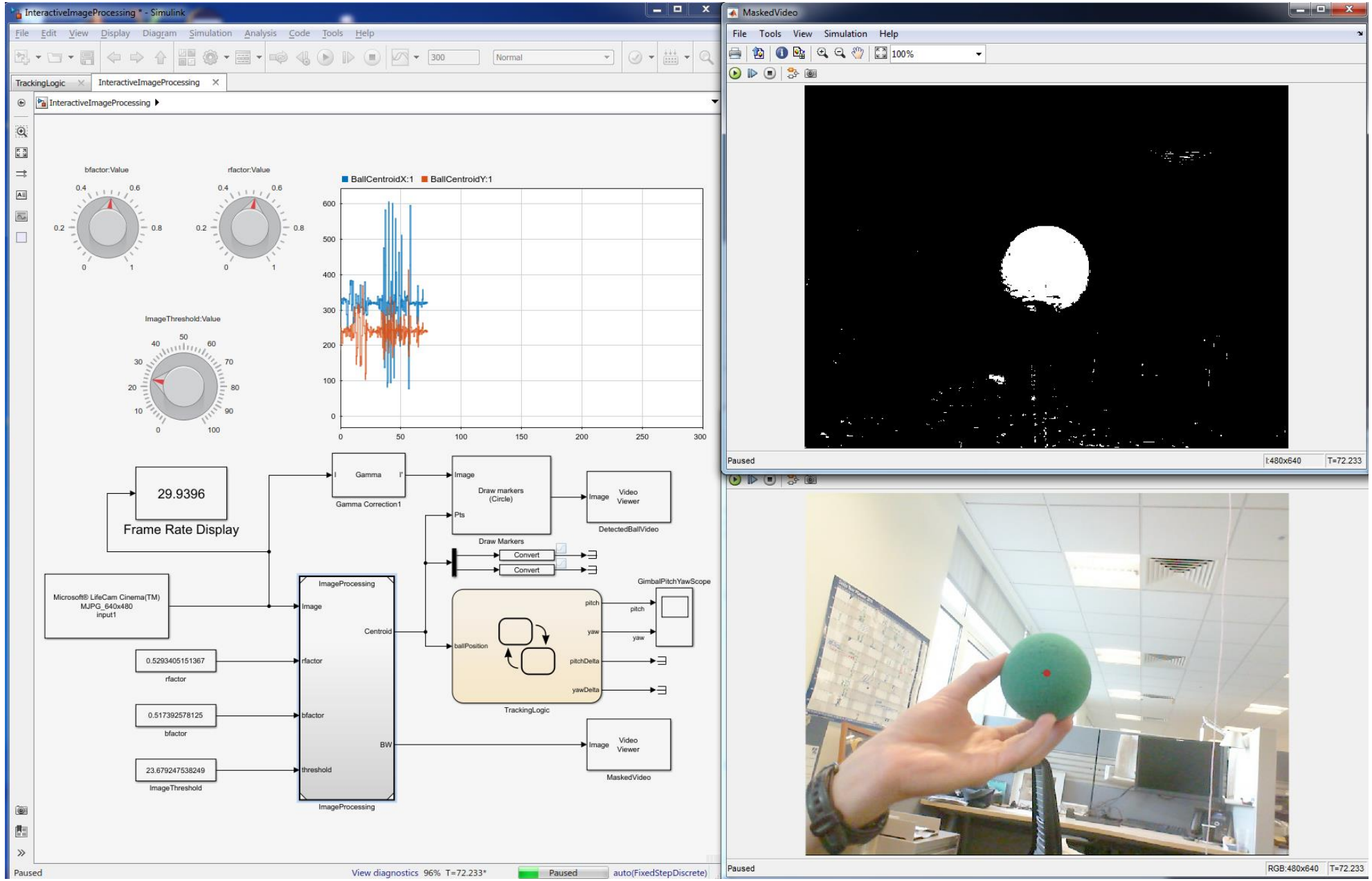
Using Simulink & Stateflow

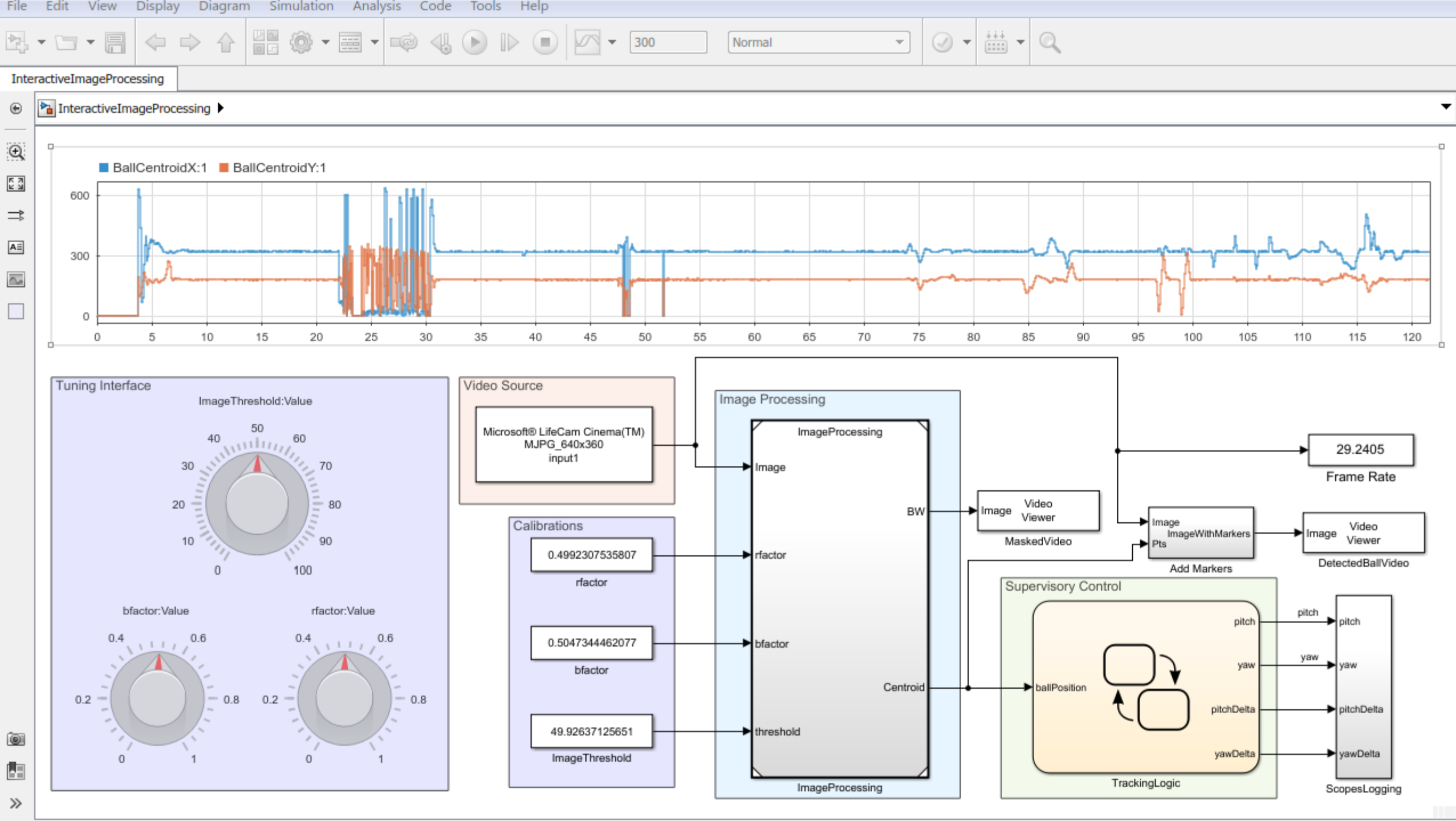
Model-Based Design Application

+

+

- Rotate a camera to track an object
- Computer vision application
- Closed-loop motor control


InteractiveImageProcessing * - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
TrackingLogic
InteractiveImageProcessing
bfactor:Value
rfactor:Value
BallCentroidX:1
BallCentroidY:1
ImageThreshold:Value
29.9396
Frame Rate Display
Gamma
Gamma Correction1
Image
Draw markers (Circle)
Pts
Draw Markers
Video Viewer
DetectedBallVideo
Convert
Convert
Microsoft® LifeCam Cinema(TM) MJPG_640x480 input1
ImageProcessing
Image
Centroid
rfactor
bfactor
threshold
BW
0.5293405151367
rfactor
0.517392578125
bfactor
23.679247538249
ImageThreshold
ballPosition
pitch
yaw
pitchDelta
yawDelta
TrackingLogic
GimbalPitchYawScope
Image
Video Viewer
MaskedVideo
ImageProcessing
Paused
View diagnostics 96% T=72.233*
Paused
auto(FixedStepDiscrete)
MaskedVideo
File Tools View Simulation Help
100%
Paused
I:480x640
T=72.233
Paused
RGB:480x640
T=72.233


File Edit View Display Diagram Simulation Analysis Code Tools Help
300
Normal
InteractiveImageProcessing
InteractiveImageProcessing
BallCentroidX:1
BallCentroidY:1
600
300
0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120
Tuning Interface
ImageThreshold:Value
0 10 20 30 40 50 60 70 80 90 100
bfactor:Value
0 0.2 0.4 0.6 0.8 1
rfactor:Value
0 0.2 0.4 0.6 0.8 1
Video Source
Microsoft® LifeCam Cinema(TM)
MJPG_640x360
input1
Calibrations
0.4992307535807
rfactor
0.5047344462077
bfactor
49.92637125651
ImageThreshold
Image Processing
ImageProcessing
Image
rfactor
bfactor
threshold
BW
Centroid
ImageProcessing
Image Video Viewer
MaskedVideo
29.2405
Frame Rate
Image
ImageWithMarkers
Pts
Add Markers
Image Video Viewer
DetectedBallVideo
Supervisory Control
ballPosition
pitch
yaw
pitchDelta
yawDelta
TrackingLogic
pitch
yaw
pitch
yaw
pitchDelta
yawDelta
ScopesLogging

What questions do we want to answer?

- Can I get the closed loop response I need?
- What current will my motor draw during operation?
- Does my system still work if component values change?
- What if…?

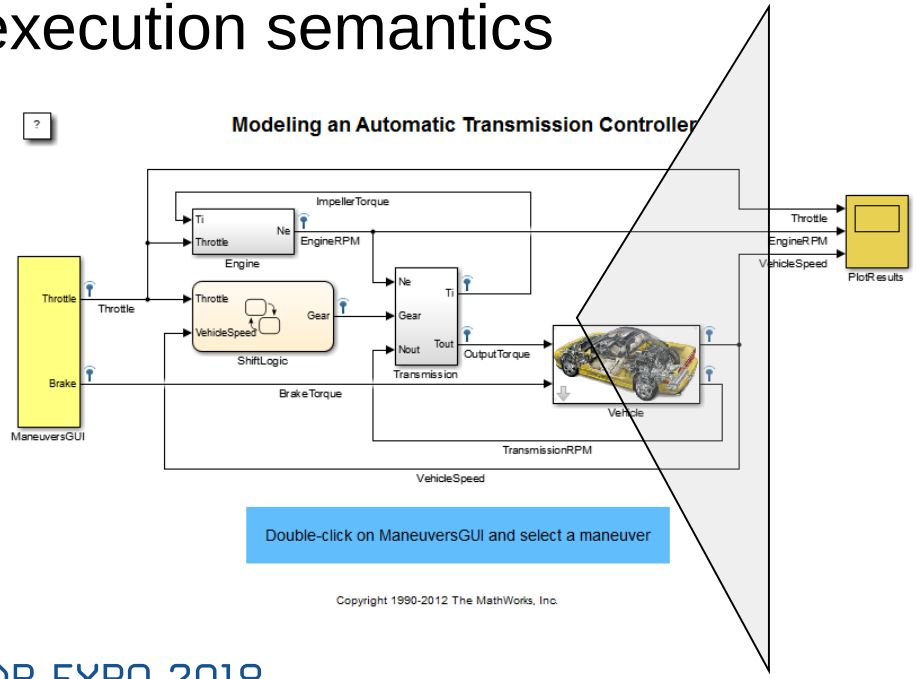
Steps in the process

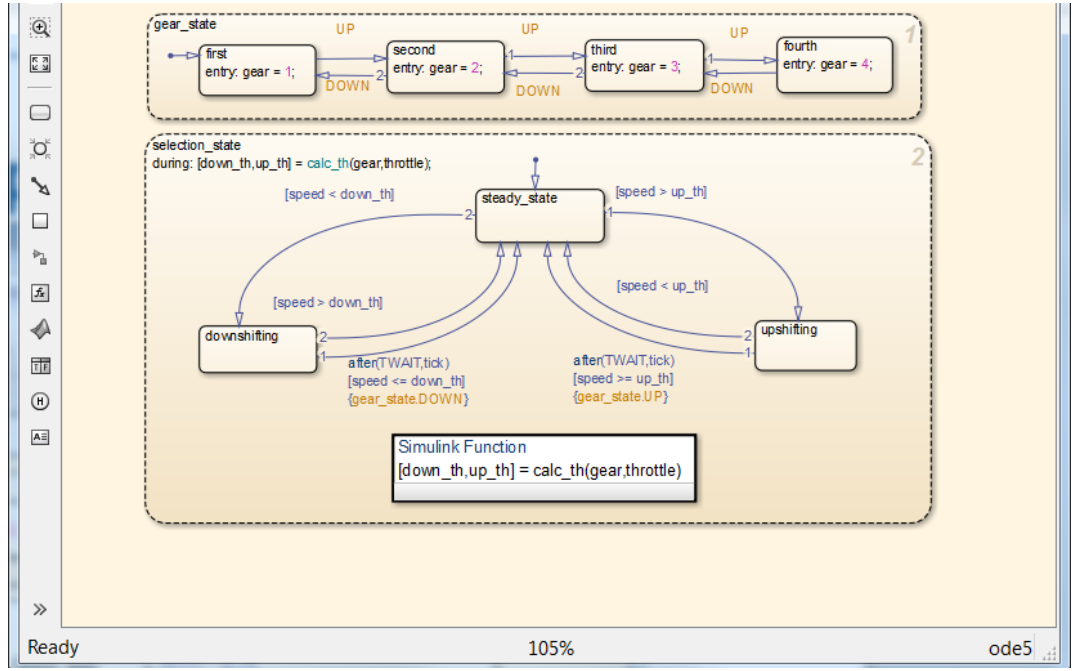
1. Model the motor
2. Model the speed controller
3. Refine the motor model using measured data
4. Model the supervisory logic
5. Validate and integrate the image processing algorithm
6. Deploy the control model to hardware

At each stage: **Simulate the model** 

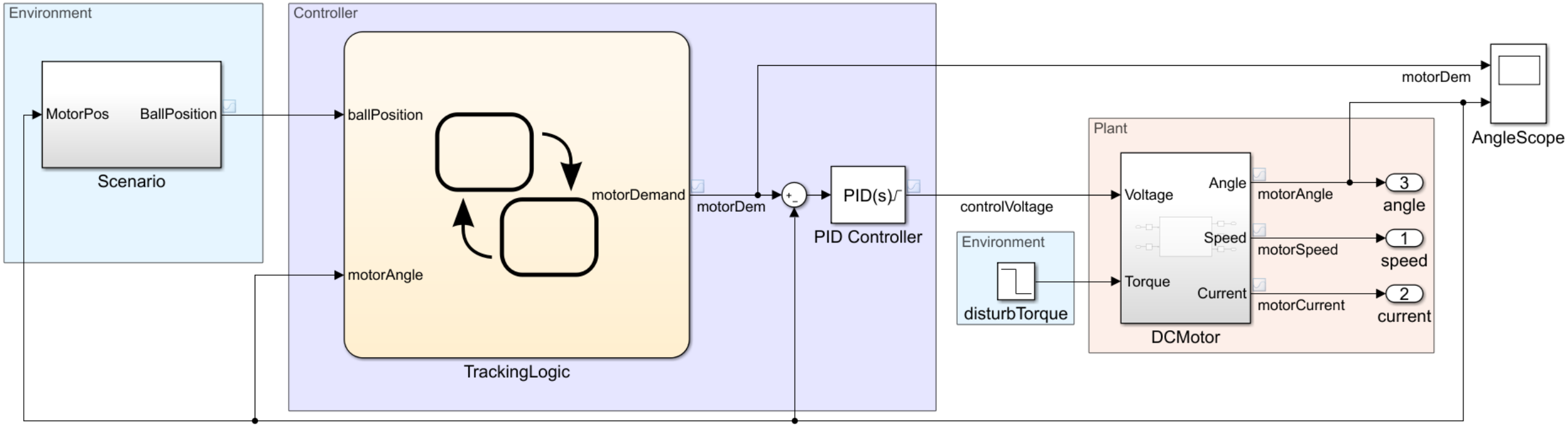
Stateflow Overview

- Extend Simulink with a design environment for developing state machines and flow charts
- Design systems containing control, supervisory, and mode logic
- Describe logic in a natural and understandable form with deterministic execution semantics





Modelling the system with Simulink and Stateflow



Conclusions

- Modelling and simulation gives you insight to make smarter decisions, earlier
- Simulink allows you to model the complete system in a single environment
- Accelerate your simulation work with the power of MATLAB