MATLAB EXPO 2017
How Simscape™ Supports Innovation for Cyber-Physical Systems
Rick Hyde
How can we use system-level modelling to support *innovative product design*?
Innovation in electric and hybrid vehicles
Innovation in robotics
Example: Quadruped running robot
Biologically-inspired design (Biomimetics)

- Animal terrestrial motion
  - Muscles are inefficient (30%)
  - Muscles are also the energy store
  - Running gait uses kinetic energy recovery
  - The leg is well modelled by a linear spring

- Innovation: Use equivalent inverted pendulum model as basis for robot
Running robot design example

Design step #1 – gait selection

- Fixed parameters
  - Leg length
  - Running speed
  - Mass

- Design parameters
  - Leg (spring) stiffness
  - Stance height

- Simple point-mass model
  - MATLAB script for trade-off
Running robot design example

Design step #2 – actuator requirements from inverse dynamics

Trajectory of Left Leg Angle

- Leg Angle
- Launch/Land
- Left Leg on Floor
- Both Legs in Air
- Right Leg on Floor
- $\alpha_{\text{launch}} = -14.95^\circ$
- $\omega_{\text{launch}} = -106.09^\circ/\text{s}$

Robot Leg Inverse Dynamics
Running robot design example

*Design step #3 – actuator selection*

- **Provide actuation force**
  - Hydraulic
    - Hydraulic motor
    - Hydraulic actuator
  - Chemical
  - Artificial muscle
  - IC engine
  - Rotary electric motor
    - Brushed
      - DC motor
      - Shunt motor
      - Series wound motor
    - Brushless
      - Induction motor
      - Servo motor (PM rotor)
      - Variable reluctance
  - Linear Electric motor
  - Linear
  - Pneumatic
    - Limited travel rotary
    - Air muscle
    - Pneumatic motor
Running robot design example

Design step #3

– actuator selection

MATLAB EXPO 2017
Running robot design example

Design step #4 – actuation validation
Running robot design example

**Design step #5 – evaluation**

Motor efficiency at rated load = 95%
Motor efficiency for trotting gait = 84%

Mechatronic Running Robot

1. Plot battery current (see code)
2. Set constraints: Planar, Planar+5DOF, 6DOF (see code)
3. Explore limb design script; Load defaults
4. Explore simulation results using sscesplore
5. Learn more about this example

MATLAB EXPO 2017

Running robot design example

*Design step automation using MATLAB scripting*

---

```matlab
%% Generate nominal gait, leg length and payload mass

% Biomechanical parameters
L = 1.0;   % Leg length (m)
m = 25;    % Mass (Kg)
k = 5315;  % Leg stiffness (N/m)

% Initial conditions for normalized positions and speeds
x0 = 0.0;  % Horizontal position of mass in middle of stance phase (m)
y0 = 0.85*L; % Height of mass in middle of stance phase (m)
u0 = 2.0;  % Horizontal speed in middle of stance phase (m/s)
```

- Automation permits greater understanding of design trade-offs
  - e.g. see effect of gearbox ratio on efficiency

<table>
<thead>
<tr>
<th>Gear ratio</th>
<th>80</th>
<th>100</th>
<th>120</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficiency</td>
<td>84%</td>
<td>81%</td>
<td>78%</td>
</tr>
</tbody>
</table>
Running robot design example

Key points

1. Multiple models
2. Each model matched to a design task
3. Design data passed between models
4. Automation to support analysis & optimisation
5. Code generation for HIL testing

Enables product design innovation in a way that starting with the CAD tool could never do
Building the right model for the task at hand can be challenging

Requirements not understood by project management

Identification of required modelling detail
Identify required modelling detail for **PMSM drives**

1. **System-level simulation**
   - Torque-speed behaviour
   - Model motor losses as part of overall efficiency calculation
   - Thermal & fault modelling

2. **Component validation**
   - Ensure motor stays within manufacturer operating limits
   - Detailed analysis of impact on other components e.g. power harmonics

3. **Component design**
   - Motor and/or drive circuitry
   - Determine overall actuation losses
   - Understand/predict fault behaviour
Building the right model for the task at hand can be challenging

Requirements not understood by project management

Identification of required modelling detail

Limited time and nothing to build on – starting from scratch

Lacking domain knowledge
Simscape library components provide a useful starting point and encapsulate some domain knowledge.
Building the right model for the task at hand can be challenging

- Requirements not understood by project management
- Identification of required modelling detail
- Limited time and nothing to build on – starting from scratch
- No data
- Lacking domain knowledge
Modelling a PMSM with limited supplier data
*Tune to measurement data*

See PMSM parameter identification example in Track 2 at 16:15pm
Using abstraction to deal with limited data

R2017a/R2017b: elec_auto_ev.slx

R2016b/R2016a/R2015b: elec_electric_vehicle.slx
Electric Vehicle Configured for HIL
1. Plot speed of vehicle (see code)
2. Plot motor torque (see code)
3. Set solver: Desktop, Real Time
4. Explore simulation results using sscexplore
5. Learn more about this example
Servomotor

This block represents a servomotor and drive electronics operating in torque-control mode, or equivalently current-control mode. The motor’s permissible range of torques and speeds is defined by a torque-speed envelope, and the output torque is assumed to track the torque reference demand \( T_r \) with time constant \( T_c \).

The servomotor block should be connected to a DC supply. Electrical losses are assumed to be the sum of a constant term plus two additional terms that are proportional to the square of the torque and the square of the speed respectively. In addition, a resistor in series with the supply can be included to model transmission losses between power supply and servomotor driver.

The block produces a positive torque acting from the mechanical C to R ports.

### Settings

**Electrical Torque**

- Parameterize losses by: Tabulated loss data
- Vector of speeds (w) for tabulated losses: rpmVec
- Vector of torques (T) for tabulated losses: trqVec
- Corresponding losses, \( P_w(T) \): LossesMat
- Supply series resistance: 0

**Electrical Losses**

- rpm

**Mechanical**

- \( N^*m \)

**Temperature Dependence**

- W

**Thermal Port**

- Ohm

[Ok] [Cancel] [Help] [Apply]
Building the right model for the task at hand can be challenging

Requirements not understood by project management

Identification of required modelling detail

Limited time and nothing to build on – starting from scratch

No data

Lacking domain knowledge
Building the right model for the task at hand can be challenging

- Requirements not understood by project management
- Identification of required modelling detail
- Limited time and nothing to build on – starting from scratch
- No data
- Lacking domain knowledge
Simscape libraries enable you to build representative models fast
Creating custom Simscape components

Example: McKibben air muscle

Steps:
1. Write out defining equations
2. Find starting point in Simscape foundation library
3. Incrementally add functionality, testing as you go
Creating custom Simscape components

**Step 1: Write out equations**

\[ L_u = \text{Un-stretched length} \]
\[ L_s = \text{Additional stretch due to force, } F \]

Assumptions:
- Volume is approximately constant
- Stretch force is proportional to \( L_s \)

Equations:
- \[ L = L_u(p) + L_s \]
- \[ F = k \times L_s \]
- \[ pV = nRT \]
Creating custom Simscape components

**Step 2: Find starting point from foundation library**

- Has equation of state
- Need to add mechanical ports & equations
Creating custom Simscape components

Step 3: Incrementally add functionality

Add:
- Two mechanical ports
- Two additional new equations

\[ L = L_u(p) + L_s \]
\[ F = k \times L_s \]

```
component air_muscle < foundation.gas.one_port_vertical
% Air Muscle (G)
% This block models a McKibben air muscle.

% Copyright 2016-2017 The MathWorks, Inc.

nodes
    H = foundation.thermal.thermal; % H:top
    R = foundation.mechanical.translational.translational; % R:bottom
    C = foundation.mechanical.translational.translational; % C:top
end
```

\[ L = L_s + L_u; \]
\[ force = K \times L_s; \]
Creating custom Simscape components

Step 3: Incrementally add functionality

Add definitions for:
- Variables
- Parameters

```matlab
variables

% Mechanical variables
force = {0, 'N'}; % Force
Ls = {0, 'm'}; % Stretch
end

parameters

K = {140, 'N/cm'}; % Stiffness
pVec = {{0 1 2 3 4 5 6}, 'bar'}; % Pressure
LuVec = {{30 27.3 25.1 23.5 22.3};
end
```
Creating custom Simscape components

Step 4: Build library and run test model
Why use Simscape?

Mechatronic Running Robot
1. Plot battery current (see code)
2. Set constraints: Planar, Planar+6DOF, 6DOF (see code)
3. Explore limb design script: Load defaults
4. Explore simulation results using ssckexplore
5. Learn more about this example

MATLAB EXPO 2017
Why use Simscape?

- Plant and control
Why use Simscape?

- Plant and control
- Multidomain
  - Electrical
  - Mechanical
  - Thermal
  - Fluid
Why use Simscape?

- Plant and control
- Multidomain
- Code generation and V&V tools
  - Test controller on HIL plant
  - Deploy to simulator
  - Use plant model in real-time controller
Why use Simscape?

- Plant and control
- Multidomain
- Code generation and V&V tools
- Libraries, examples, documentation & webinars
Why use Simscape?

- Plant and control
- Multidomain
- Code generation and V&V tools
- Libraries, examples, documentation & webinars
- **Simscape language – build custom components**

```
21  parameters
22    K = [140, 'N/cm'];  % Stiffness
23    pVec = [0 1 2 3 4 5 6], 'bar';  % Pressures
24    LuVec = [30 27.3 25.1 23.5 22.3 21.7 21.5], 'cm';
25  end
```
Why use Simscape?

- Plant and control
- Multidomain
- Code generation and V&V tools
- Libraries, examples, documentation & webinars
- Simscape language – build custom components

Workflow
  - Tight integration with MathWorks control and optimization tools
  - MATLAB for scripting and automation
  - Fault-capable components (R, L, C, Servomotor, …)
Why use Simscape?

- Plant and control
- Multidomain
- Code generation and V&V tools
- Libraries, examples, documentation & webinars
- Simscape language – build custom components
- Workflow
  - Tight integration with MathWorks control and optimization tools
  - MATLAB for scripting and automation
  - Fault-capable components (R, L, C, Servomotor, …)
- Support, training, consulting
- MATLAB Central
How to find out more

- MathWorks physical modelling page:  

- Steve Miller’s introduction video  

- MATLAB Central File Exchange  
  - https://www.mathworks.com/matlabcentral/fileexchange/

- Contact us direct