Motor Control with Arduino

A Case Study in Data-Driven Modeling and Control Design

Toledo, 22-II-2013
Why Data-Driven Control?

- Two approaches to modeling
  - **First-principles**: requires knowledge of math / physics of the system
  - **Data-driven**: requires measured input-output data

- Modeling from first-principles can get challenging
Demo: DC Motor Controller using Arduino Uno

![Diagram of DC Motor Controller using Arduino Uno](image)

**Step Response**

<table>
<thead>
<tr>
<th>Time (seconds)</th>
<th>Angle (Deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
</tr>
</tbody>
</table>

- Reference
- Simulation
- Hardware
Workflow

1. Data Acquisition

Hardware
Workflow

1. Data Acquisition

Hardware

Datasets

2. System Identification

Controller Design

Data Acquisition

Plant Model

Real-Time Testing

Workflow
Workflow

1. Data Acquisition

2. System Identification

3. Controller Design

Hardware

Datasets

Plant Model
Workflow

1. Data Acquisition
   - Hardware
   - Datasets
   - System Identification
   - Controller Design
   - Plant Model

2. Real-Time Testing
   - Control System
Agenda

- Data Acquisition
  - Hardware setup
  - Run on Target Hardware

- System Identification
  - Linear model estimation
  - Nonlinear model estimation

- Controller Design
  - PID controller tuning
  - Desktop simulation with a nonlinear model

- Real-Time Testing and Controller Implementation
  - Deployment to Arduino Uno
  - Real-time controller evaluation
DATA ACQUISITION
Hardware Setup

Host Computer

Arduino Uno

DC Motor

Serial communication

Motor driver
Run on Target Hardware

A Simulink feature in R2012a that:

- Creates an executable file from a model, and runs it on target hardware

- Is available from the model’s Tools menu
  - Tools > Run on Target Hardware

- Uses a Target Installer to install support packages for specific target hardware

Requires Simulink®
Hardware Setup

Host Computer

Arduino Uno

DC Motor

Serial communication (using Run On Target Hardware)

Motor driver
SYSTEM IDENTIFICATION
Model Fidelity vs. Cost

- Model the dynamics that matter for your analysis
- Balance cost and model fidelity
Linear System Identification

- Create continuous-time transfer functions from data
- Use either time or frequency domain data containing an arbitrary number of inputs and outputs
- Estimate other linear models: state-space, process models, and parametric models

Requires System Identification Toolbox™
Hammerstein-Wiener structure

\[ y(t) = u(t) \cdot \frac{B(q^{-1})}{F(q^{-1})} \cdot z(t) \]
Example of NLARX model

\[ y(t) = f[y(t - 1), y(t - 2), y(t - 3), u(t - 1), u(t - 2)] + e(t) \]
Nonlinear System Identification

- Estimate Hammerstein-Wiener models and nonlinear ARX models
- Use a variety of dynamic nonlinearities such as wavelets, neural networks, piecewise linear, etc.
- Estimate signal saturation and dead-zone behaviors affecting linear systems
- Use custom regressors in nonlinear ARX models for greater flexibility in modeling

Requires System Identification Toolbox™
CONTROLLER DESIGN
Automatic PID Tuning

- Automatically linearizes Simulink models and finds gain values to meet specifications
- Provides additional fine-tuning capability with simple sliders

Requires Simulink Control Design™

For more info, watch our webinar “PID Control Made Easy”.
REAL-TIME TESTING AND CONTROLLER IMPLEMENTATION
Real-Time Testing

Host Computer

Arduino Uno

DC Motor

Serial communication
(Using Run On Target Hardware)

Motor driver
Workflow

1. Data Acquisition
2. System Identification
3. Controller Design
4. Real-Time Testing

Hardware

Datasets

Control System

Plant Model
Summary

- MATLAB and Simulink support data-driven control design

- MATLAB and Simulink provide an environment for
  - Data acquisition
  - System identification
  - Control design
  - Real-Time testing