Design and Prototype Real-Time DSP Systems with MATLAB

Youssef Abdelilah  Houman Zarrinkoub
Agenda

- Create a real-time simulation test bench
- Develop real-time DSP algorithms in MATLAB
- Generate C/C++ Code to accelerate simulation
- Example
  - Acoustic tracking system using Microsoft ® KINECT ™
Streaming data is everywhere!

Aero/Defense

Radar

Wireless and Mobile

Sensors

Medical

Consumer Audio
Challenges of real-time DSP system design

Real-Time Simulations

“I have to process large data and test my simulations with streaming signals. I need a simulation testbench that can keep up with real-time data.”

Rapid Innovation

“I need to find innovative algorithms and create a working system prototype very quickly.”

Simulation acceleration

“I need to optimize my high-level MATLAB algorithm for speed. I then need to verify that the optimized code works the same way as the original MATLAB code.”
What MATLAB & DSP System Toolbox offer

- **Real-Time Simulations**
  - Streaming techniques and hardware peripheral access that speed up simulations and reduce memory footprint

- **Rapid Innovation**
  - Pre-defined algorithms that enable quick prototyping

- **Simulation acceleration**
  - Support for C/C++ code generation that enable design continuity and faster simulation
What will you see today?

**Acoustic Tracking Example**

1. Test bench for processing streaming data
   - Easy connectivity to hardware in MATLAB

2. Integrate algorithm with test bench
   - Verify real-time application

3. Speed up simulation
   - Use automatic C/C++ code generation
Acoustic tracking example

Sensor Device (KINECT)

Sound source

Algorithm

delays

d
Part 1: Test bench and peripheral access

Acoustic Tracking Example

1. Test bench for processing streaming data
   - Easy connectivity to hardware in MATLAB

2. Integrate algorithm with test bench
   - Verify real-time application

3. Speed up simulation
   - Use automatic C/C++ code generation
Streaming data

- Streaming techniques process continuous data by breaking it up into “frames”
  - Memory efficient

- Streaming algorithms in DSP System Toolbox provide
  - Implicit data buffering, state management and indexing
  - Simulation speed-up by reducing overhead
Streaming test bench in MATLAB

```matlab
% Visualization of audio spectrum frame by frame
Microphone   = dsp.AudioRecorder;
Speaker      = dsp.AudioPlayer;
SpecAnalyzer = dsp.SpectrumAnalyzer;
 tic;
while (toc < 30)
    audio = step(Microphone);
    step(SpecAnalyzer,audio);
    step(Speaker, audio);
end
```

Default Sound Card

Spectrum Analyzer
% Visualization of audio spectrum frame by frame
Microphone = dsp.AudioRecorder;
Microphone.DeviceName = 'Microphone Array (Microsoft Kinect USB Audio)';
Speaker = dsp.AudioPlayer;
SpecAnalyzer = dsp.SpectrumAnalyzer;
tic;
while (toc < 30)
    audio = step(Microphone);
    step(SpecAnalyzer,audio);
    step(Speaker, audio);
end
Part 2: Algorithm and integration

Acoustic Tracking Example

1. Test bench for processing streaming data
   - Easy connectivity to hardware in MATLAB

2. Integrate algorithm with test bench
   - Verify real-time application

3. Speed up simulation
   - Use automatic C/C++ code generation
Algorithm design

\[ \theta = f(d) \]

Algorithm

dsp.FIRInterpolator
dsp.Crosscorrelator

Median

Delay to angle of Arrival

\[ \theta = f(d) \]
Algorithm design with DSP System Toolbox

% Create and initialize objects
Interpolator = dsp.FIRInterpolator('Numerator',b);
Correlator = dsp.Crosscorrelator;

% Process audio data in a loop
for k = 1:nFrames
    inIntp = step(Interpolator, in);
    xc = step(Correlator, inIntp(:,1), inIntp(:,2));
    [~,Index] = max(xc);
End
Integrate & verify the algorithm

% Visualization of audio spectrum frame by frame
Microphone = dsp.AudioRecorder;
Microphone.DeviceName = 'Microphone Array (Microsoft Kinect USB Audio)';
Speaker = dsp.AudioPlayer;
SpecAnalyzer = dsp.SpectrumAnalyzer;
tic;
while (toc < 30)
    audio = step(Microphone);
    [delays, xCorrelator, thetas] = Algorithm(audio);
    step(SpecAnalyzer, audio);
    step(Speaker, audio);
end
Acoustic tracking example
Part 3: Accelerate simulation

Acoustic Tracking Example

1. Test bench for processing streaming data
   - Easy connectivity to hardware in MATLAB

2. Integrate algorithm with test bench
   - Verify real-time application

3. Speed up simulation
   - Use automatic C/C++ code generation
Stream processing:
Data acquisition & algorithm times

As long as

Data acquisition + Algorithm processing $\leq$ Frame time

We have

Real-time signal processing
Acceleration techniques

- Pre-allocation and vectorization
- Pre-defined efficient implementations of algorithms
- Generate MEX files automatically with MATLAB Coder
- Parallel computations on multicore computers, GPUs, and clusters
MATLAB to C code generation

MATLAB Coder

```
function [delay, xc] = AcousticTracking(u,k)
  %#codegen
  interpFactor=8;
  L=64;
  idx=(1:L);
  persistent interpolator xCorrelator
  if isempty(interpolator)
    %design FIR interpolator filter
    b = interpFactor * fir1((2*interpFactor*8-1),1/interpFactor);
    %create an Interpolator System object
    interpolator=dsp.FIRInterpolator(...
      'InterpolationFactor',interpFactor,...
      'Numerator',b);
  end
  %create Cross correlator System object
  xCorrelator = dsp.Crosscorrelator('Method', 'Frequency Domain');
end
```

Algorithm.m

Algorithm.mex*
Performance assessment

- Effect of code generation on simulation speed
  - MEX Functions: automatically generated C code

<table>
<thead>
<tr>
<th>Algorithm version</th>
<th>Elapsed time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline function</td>
<td>2.3594 (sec)</td>
</tr>
<tr>
<td>MEX function</td>
<td>0.4979 (sec)</td>
</tr>
</tbody>
</table>

- Acceleration ratio of about **5 x**
- Extent of acceleration depends on the type of algorithm
DSP System Toolbox

Over 300 algorithms for
- Advanced Filter Design, Adaptive Filters, Multi-rate Filters
- FFT, DCT and other Transforms
- Stream processing, time and frequency-domain visualization

Algorithm libraries in MATLAB

Algorithm libraries in Simulink
Summary

- **DSP System toolbox** provides
  - **Streaming techniques** for real-time signal processing
  - Easy access to peripheral hardware
  - Efficient **pre-defined algorithms** for design prototyping

- **Simulation acceleration** by
  - Automatic **MATLAB to C/C++** code generation (MEX functions)
  - Enabled by **MATLAB Coder**