POSTER P6312 | CONTACT NAME Harsh Purohit: harsh.purohit88@gmail.com

Parallel computation of the value set of frequency response for uncertain systems with GPGPU

Harsh Purohit and P.S.V. Nataraj
GPU Center Of Excellence
Indian Institute of Technology, Bombay, India

Introduction

In reality, exact mathematical modeling for any engineering system (e..g. Fig 1) is impossible. The main reasons for this difficulty are parameter variations and high frequency noise. But we can tackle this issue with the computation of the value set of the frequency response.

What is the value set of frequency response?
The value set of frequency response can be obtained by the calculation of the magnitudes and phases for all possible uncertain transfer functions.

Many robust control techniques have been evolved for designing optimal control system in the presence of uncertainty for instance quantitative feedback theory (QFT)[1]. Such techniques use value set of frequency response in the design procedure.

Figure.1. Different engineering systems. Helicopter, magnetic levitation setup and industrial emulator (clockwise)

These value sets of frequency response are used to calculate the frequency bounds that express the closed loop control specification like stability and performance. Afterwards, the loop shaping or the controller synthesis becomes very straightforward.

Motivation

The most widely used method for generating value sets is to grid the parameter set and calculate the transfer function values at each parameter point[2]. This approach is computational intensive for the systems with higher number of parameters (e.g. 3-DOF longitudinal transfer function of aircraft)

For example, a plant with 'm' uncertain parameter using 'N' gridding points transfer function values needs to be calculated 'N^m' times. But this evaluation is independent to each other and it motivates us to calculate it in parallel. Here, we are proposing an algorithm for parallel computation of value set of frequency response which can be easily implemented using GPGPU with Matlab .

Method

In general, any system can be represented by its transfer function (relationship between input and output),

$$P(s) = \left\{ \frac{k\prod i(s+z_i)}{\prod j(s+p_j)} \right\}$$

Where,

$$z_i \in [z_{imin}, z_{imax}] \quad p_j \in [p_{jmin}, p_{jmax}]$$

P(s) represents a family of plants rather than by a standalone expression. Magnitude and phase expression for the given family of plant with sampling grid of 'N' points can be described as follows

$$\text{Magnitude} = \left|P(j\omega)\right|_{s=j\omega} \forall z_i^1, z_i^2 \ldots z_i^N, p_j^1, p_j^2 \ldots p_j^N$$

$$\text{Phase} = \angle P(j\omega)_{s=j\omega} \forall z_i^1, z_i^2 \ldots z_i^N, p_j^1, p_j^2 \ldots p_j^N$$

Objective

First grid the parameter interval into N points and for each point in the mentioned grid find the magnitude and phase at a given frequency.

Algorithm for parallel implementation

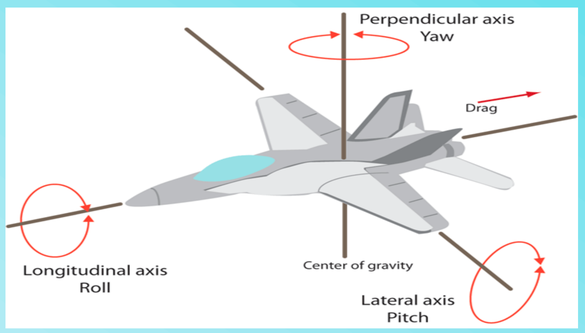
Algorithm 1: Template generation

Input: (i) Plant transfer function $\mathcal{P}(p_1, p_2, p_3, \ldots p_m, \omega)$ where p_i are plant parameters and ω is frequency
(ii) v_i = % variation of parameter p_i
(iii) N = Number of samples for a given parameter vector
(iv) Design frequency range $\Omega := [\omega_l, \omega_h]$
Output: (i) Magnitude ($\mathfrak{M}$) and Phase ($\mathfrak{P}$) vectors for all possible plant transfer functions
(ii) Nichols plot
Procedure;

1 $\mathfrak{I}_i \leftarrow [\, a_i := p_i - \frac{v_i * p_i}{100},\ p_i + \frac{v_i * p_i}{100} =: b_i] \quad \forall i \in \{1, 2, \ldots m\}$ ▷ Initialize intervals with variation in parameters
2 $\mathfrak{G}_i \leftarrow \{\, a_i + s * k \mid s = \frac{b_i - a_i}{N};\ k \in \{1, 2, \ldots N\}\}$
where $\mathfrak{G}_i \in \mathfrak{I}_i \ \forall i \in \{1, 2, \ldots m\}$ ▷ Generate linear grid from each parameter interval $\mathfrak{I}_i$
$\mathfrak{G}_\omega \leftarrow \{e^{(ln(\omega_l) + s*k)} \mid s = \frac{ln(\omega_h) - ln(\omega_l)}{N};\ k \in \{1, 2, \ldots N\}\}$
where $\mathfrak{G}_\omega \in \Omega$ ▷ Generate logarithmic grid from frequency interval Ω
$\mathfrak{G}^{m+1} := \{\mathfrak{G}_1, \mathfrak{G}_2, \ldots \mathfrak{G}_m, \mathfrak{G}_\omega\}$
3 **for** $\forall i \in \{1, 2, \ldots m+1\}$ ▷ Generate all possible permutation from parameter grid vectors
$\Upsilon_i := \{Y_i, Z_i\}$
where $Y_i \leftarrow \{\mathfrak{G}_i, \mathfrak{G}_i \ldots (i-1) copies\}$
$Z_i \leftarrow \{\mathfrak{g}_1^i, \mathfrak{g}_2^i \ldots \mathfrak{g}_N^i\}$
where $\mathfrak{g}_j^i \leftarrow \{\alpha_j^i, \alpha_j^i \ldots (m-i+1) copies\} \quad \forall \alpha_j^i \in \mathfrak{G}_i$
4 **end**
$\mathfrak{M} \leftarrow \text{magnitude}(\mathcal{P}(\Upsilon_i))$ ▷Vecorize computation with magnitude expression of the given plant
$\mathfrak{P} \leftarrow \text{Phase}(\mathcal{P}(\Upsilon_i))$ ▷Vecorize computation with phase expression of the given plant
5 *Nicholsplot* $(\mathfrak{M}, \mathfrak{P})$ ▷Plot magnitude(in dB) and phase (Deg)

Case study

Basic 3-DOF longitudinal aircraft model has been depicted in the following.



3 –DOF longitudinal aircraft model for input force to speed is considered as follows[3, page338] . The possible parameter variation is also shown

$$P(s) = \frac{As^2 + Bs + C}{Hs^4 + Is^3 + Js^2 + Ks + L}$$

$$A \in [4.4184, 8.2056]; B \in [-3448.9, -6405.1]; C \in [-3011.4, -5592.6]$$

$$H \in [473.1300, 878.6700]; I \in [9597, 1782.3]; J \in [3821.3, 7096.5];$$

$$K \in [60.41, 112.20]; L \in [31.34, 58.21]$$

Frequency range is chosen as 1 to 60 rad/sec and grid size is variable to measure the performance. Platform for this experiment is used as follows:
Intel's processor i5 3.3GHz, 4 GB Ram, Nvidia K 40 GPU, Matlab 2015a (with PCT toolbox), CUDA toolkit 7.5, Windows 7

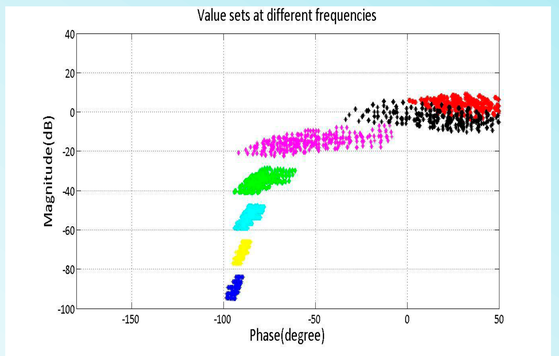


Figure.2. Plot for value sets at different frequencies. (Different colors shows different frequencies)

Figure 2 shows magnitude and phase plot (Nichols plot) for the given aircraft transfer function. We have chosen seven different frequency samples and it is very clear that at lower frequency range, parameter variation effect is much higher than at high frequency range.

Samples(N)	Total Number of plants to compute	Simple serial for loop time (sec)	Vectorize implementation time (sec)	GPGPU time (sec)
1	1	0.0284	0.0344	0.1491
2	512	0.1345	0.0793	0.1492
3	19683	3.4404	0.1040	0.1889
4	262144	38.9782	0.5124	0.1657
5	1953125	233.2252	3.3061	0.3144
6	10077696	1173.993	21.2070	1.2569
7	40355623221	5645.0215	124.2143	3.2523

Table 1. Comparison table for serial, vectorize and GPGPU implementation with different size of samples.

Conclusion and Future Scope

We have obtained the speedup of up to 40 times (in comparison with vectorize matlab code) with GPGPU in comparison to CPU computation. It is concluded that GPU computation is very useful for higher number of uncertain parameters and intense grid size in value set generation. In future, we are planning to implement bound generation step used in designing controllers on GPGPU.

References

[1] I. M. Horowitz, Quantitative Feedback Design Theory (QFT)., Boulder Colorado, QFT Publications.

[2] P. S. V. Nataraj and G. Sardar, "Template generation for continuous transfer functions using IA" Automatica, vol. 36, pp. 111–119, 2000.

[3] Thomas R. Yechout, Steven L. Morris, David E. Bossert, Wayne F. Hallgren, " Introduction to Aircraft Flight Mechanics: Performance, Static Stability, Dynamic Stability, and Classical Feedback Control", American Institute of Aeronautics and Astronautics, 2003.