IMPLEMENTING ADAPTIVE BEAMFORMING ON THE GPU FOR REAL TIME ULTRASOUND IMAGING

J. P. Åsen[1], J. I. Buskenes[2], C.-I. C. Nilsen[2], A. Austeng[2] and S. Holm[1,2]

[1]Mi Lab, Norwegian University of Science and Technology, Trondheim, Norway
[2]Department of Informatics, University of Oslo, Oslo, Norway

ABSTRACT

1. INTRODUCTION

Introduction to Capon adaptive beamforming, ultrasound imaging and GPU.

Delay and sum beamformer (DAS):

$$z[n] = \sum_{m=0}^{M-1} \mathbf{w}_m^* \mathbf{x}_m[n - \Delta_m[n]] = \mathbf{w}^H \mathbf{x}[n], \qquad (1)$$

where M is the number of elements.

Capon weights:

$$\mathbf{w}[n] = \frac{\mathbf{R}[n]^{-1}\mathbf{a}}{\mathbf{a}^H \mathbf{R}[n]^{-1}\mathbf{a}}, \qquad (2)$$

where $\mathbf{a} = \mathbf{1}$ when $\mathbf{x}$ are pre-delayed.

Estimation of $\mathbf{R}[n]$: $\hat{\mathbf{R}}[n] = \mathbf{x}[n]\mathbf{x}[n]^H$.

In order to get a well conditioned $\hat{\mathbf{R}}$, avoid signal cancellation and DAS-like speckle, $\hat{\mathbf{R}}$ has to be averaged over L subarrays and N_{avg} time samples.

$$\hat{\mathbf{R}} = \frac{\sum_{n_{avg}=-N_{avg}}^{N_{avg}} \sum_{l=0}^{M-L+1} \mathbf{x}_l[n + n_{avg}]\mathbf{x}_l[n + n_{avg}]^H}{(2N_{avg}+1)(M-L+1)}, \qquad (3)$$

where $\mathbf{x}_l$ is the l^{th} subarray $[x_l[n], \ldots x_{l+L}[n]]$.

2. METHOD

2.1. Building the Sample Covariance Matrix

Building R using a sliding window approach across the subarrays. We have a limited amount of fast, near-core memory on the GPU. On NVIDIA architecture this memory is known as shared memory, and is restricted to 48KB per compute block. The maximum block size of 32x32, further restrict how large arrays we can handle inside a single compute block.

2.2. Solving Small Linear Systems

Gauss Jordan elimination to solve $\mathbf{R}^{-1}\mathbf{1}$.

3. RESULTS

Present images before and after Capon weights has been applied. Present graph showing running times for Cython-Capon v.s. CUDA-Capon for different choices of parameters.

4. DISCUSSION

5. CONCLUSION