Efficient Computation of Spherical Harmonic Transform using Parallel Architecture of CUDA

Weiyu Huang, Zubair Khalid and Rodney A. Kennedy
Research School of Engineering, College of Engineering and Computer Science,
The Australian National University, Canberra, Australia.
Email: {zubair.khalid, rodney.kennedy}@anu.edu.au

***Abstract*—Spherical harmonics serve as basis functions on the unit sphere and spherical harmonic transform is required in analysis and processing of signals in the spectral domain. We investigate the possibility of parallel computation of spherical harmonic transform using Compute Unified Device Architecture (CUDA) with no communication between parallel kernels. We identify the parallel components in the widely used spherical harmonic transform method proposed by Driscoll and Healy. We provide the implementation details and compare the computational complexity with the sequential algorithm. For a given bandlimited signal with maximum spherical harmonics degree L, using the $O(L)$ number of parallel processing kernels, we present that the spherical harmonic coefficients can be calculated in $O(L\log^2 L)$ time as compared to $O(L^2\log^2 L)$. For corroboration, we provide the simulation results using CUDA which indicate the reduction in computational complexity**

I. Introduction

There has been growing interest in developing and extending the signal processing techniques to the non-Euclidean spaces such as unit sphere, as it has direct applications in various branches of physical sciences and engineering, such as geology [1], cosmology [2], computer graphics [3], medical imaging [4, 5] and wireless communication systems [6]. A fundamental analogy is the existence of spherical harmonics which corresponds to Fourier transform and serve as basis functions for signals on the unit sphere. The spherical harmonic transform is a neceassary step towards frequency analysis on the sphere. However, the transform lacks a fast algorithm, as the computational complexity of the direct computation using the definition is $O(L^3)$, where L denotes the maximum spherical harmonic degree of a given signal. Much research have been done to improve the computational complexity, among which the most widely used efficient algorithm is presented by Driscoll and Healy [7] that reduces the overall complexity to $O(L^2\log^2 L)$.

With the rapid advances in Graphics Processing Units (GPUs) and its current easy access to perform computations, parallel processing in hardware have been used to accelerate many non-graphical area, including biology, cryptography and other fields [8–10]. The large input data size and the high maximum spherical harmonic degree required in practical usage, makes it attractive to explore computing of the the spherical harmonic transform in parallel. Compute Unified Device Architecture (CUDA) is one of the most commonly used parallel computational architectures developed by Nvidia, which enables parallel computation on GPUs. Research work has been done to parallelize and accelerate the spherical harmonic transform. For example, Inda et al. [11] uses fast Chebyshev transform and bulk synchronous parallel model to make the overall computational complexity reduce to $4.25\frac{L^2}{p}\log^2 L$, where p is the number of processors and is assumed as L. However, there are many data transfers between different processors, which makes it slower in practical implementation and the alternate method with no communication between different processors is given by [12] but it took the direct transform using classical spherical harmonics into consideration rather than the relatively efficient Driscoll and Healy algorithm.

In this work, we evaluate the parallel implementation of Driscoll and Healy spherical harmonic transform algorithm [7] which is widely used to calculate the spherical harmonic transform for a given bandlimited signal. We provide the survey of existing literature on efficient spherical harmonic transform methods. We classify the independent sequential operations in the existing method [7] and present the modified algorithm which is suitable to run on the parallel architecture. For a given badlimited signal with maximum spherical harmonics degree L, the modified algorithm uses the $O(L)$ number of processors and has a reduced computational complexity $O(L\log^2 L)$ as compared to $O(L^2\log^2 L)$. We provide an implementation procedure and present a comparison of theoretical computational complexities. We simulate the modified algorithm using CUDA and show that the parallel algorithm performs better for maximum spherical harmonic degree greater than for a specific spherical harmonics degree, which is hardware architecture dependent.

This paper is organized as follows. In section II, we give the mathematical background, define the spherical harmonic transform operation, survey the relevant research work and provide a brief introduction to CUDA. We revisit the sequential algorithm [7], present the modified parallel algorithm section III and discuss computational complexity. The experimental result is shown in Section IV, and Section V concludes the paper.

II. Problem Formulation

In this section, we first present the mathematical background related to signals defined on the unit sphere and spherical harmonics. Later, we present spherical harmonic transform,

978-1-4577-1180-0/11/$26.00

followed by the literature review of fast spherical harmonics and an introduction to Compute Unified Device Architecture (CUDA) which will be used in this work for parallel implementation.

A. Mathematical Background - Signals on the Unit Sphere

On the unit sphere, $L^2(\mathbb{S}^2)$ is the domain of square integrable functions $f(\hat{\boldsymbol{x}}) = f(\theta,\phi)$ where $(\hat{\boldsymbol{x}}) = (\theta,\phi) \triangleq (\sin\theta\cos\phi,\ \sin\theta\sin\phi,\ \cos\theta)' \in \mathbb{R}^3$ is the unit vector, $\theta \in [0,\pi]$ denotes the co-latitude and $\phi \in [0,2\pi)$ denotes the longitude. θ is measured with respect to the positive z-axis and ϕ is measured with respect to the positive x-axis in the $x-y$ plane. The inner product on $L^2(\mathbb{S}^2)$ is defined as

$$\langle f, g\rangle \triangleq \int_{\mathbb{S}^2} f(\hat{\boldsymbol{x}})\overline{g(\hat{\boldsymbol{x}})}ds(\hat{\boldsymbol{x}}) \tag{1}$$

where, $f, g \in L^2(\mathbb{S}^2)$, $ds(\hat{\boldsymbol{x}}) = \sin\theta d\theta d\phi$ and the integration is carried out over the unit sphere.

The spherical harmonics are orthonormal basis functions and spherical harmonic coefficients serve as spectral domain for signals defined on the unit sphere. The spherical harmonics, $Y_\ell^m(\theta,\phi)$, for degree $\ell \geq 0$ and order $|m| \leq \ell$ are defined as [13]

$$Y_\ell^m(\theta,\phi) = N_\ell^m P_\ell^m(\cos\theta)e^{im\phi} \tag{2}$$

where $i = \sqrt{-1}$ is the imaginary unit, $P_\ell^m(.)$ are the associated Legendre polynomials and are defined for $m \geq 0$ as

$$P_\ell^m(x) = \frac{(-1)^m}{2^\ell \ell!}\sqrt{(1-x^2)^m}\frac{d^{\ell+m}}{dx^{\ell+m}}(x^2-1)^\ell \tag{3}$$

$$P_\ell^{-m}(x) = (-1)^m\frac{(\ell-m)!}{(\ell+m)!}P_\ell^m(x) \tag{4}$$

where $|x| \leq 1$ and N_ℓ^m is the normalization constant

$$N_\ell^m = \sqrt{\frac{2\ell+1}{4\pi}\frac{(\ell-m)!}{(\ell+m)!}} \tag{5}$$

and is chosen such that the spherical harmonics form an orthonormal set of basis functions for $L^2(\mathbb{S}^2)$ with inner product defined as $\langle Y_\ell^m, Y_{\ell'}^{m'}\rangle = \delta_{\ell\ell'}\delta_{mm'}$. Thus, any function $f(\theta,\phi)$ on $\mathbb{S}^2$ can be expressed using spherical harmonics as basis functions

$$f(\hat{\boldsymbol{x}}) = \sum_{\ell=0}^{\infty}\sum_{m=-\ell}^{\ell} f_\ell^m Y_\ell^m(\hat{\boldsymbol{x}}) \tag{6}$$

where f_ℓ^m is the spherical harmonic coefficient and can be defined as an inner product of $f(\hat{\boldsymbol{x}})$ and $Y_\ell^m(\hat{\boldsymbol{x}})$

$$f_\ell^m \triangleq \langle f, Y_\ell^m\rangle = \int_{\mathbb{S}^2} f(\hat{\boldsymbol{x}})\overline{Y_\ell^m(\hat{\boldsymbol{x}})}ds(\hat{\boldsymbol{x}}) \tag{7}$$

Also, the spherical harmonic coefficients of a real signal satisfy

$$\overline{f_\ell^m} = (-1)^m f_\ell^{-m} \tag{8}$$

B. Spherical Harmonic Transform

The spherical harmonic transform of a given bandlimited signal $f(\theta,\phi)$ with maximum spherical harmonics degree L is defined as the calculation of all the spherical harmonic coefficients f_ℓ^m for all $0 \leq \ell \leq L$ and $0 \leq |m| \leq \ell$. Each spherical harmonic coefficient f_ℓ^m is obtained by projecting the signal $f(\theta,\phi)$ onto the spherical harmonics $Y_\ell^m(\theta,\phi)$ as given in (7). Numerically, the integration involved in spherical harmonic transform over the whole unit sphere is carried out as summation over the number of sample points on the sphere. Equal area sampling on the unit sphere results in the number of points on the sphere which are equidistant from each other [14]. The drawback of this type of sampling is that we will not have an independent control over latitude θ and longitude ϕ. We can have uniform and equiangular grid sampling of a signal on the unit sphere over the latitude and longitude, which keeps the independence between θ and ϕ for fix value of θ, but the number of samples are denser near the poles. Since, we evaluate the integration as summation, therefore, there must be a weighting factor which depends on θ and normalizes the effect of dense sampling. We present the result in [7] (Theorem 3) that relates the number of sample points needed to accurately determine the spherical harmonic coefficients of the bandlimited signal.

Theorem 1: If $f(\theta,\phi)$ denotes the bandlimited signal on the unit sphere with maximum spherical harmonics degree L, we only need a equiangular grid $2b \times 2b$ samples over $\theta \in [0,\pi]$ and $\phi \in [0,2\pi)$ on the sphere to determine all of the spherical harmonic coefficients f_ℓ^m of degree $0 \leq \ell \leq L$ and all orders $-\ell \leq m \leq \ell$, *i.e.*,

$$f_\ell^m = \frac{\sqrt{2\pi}}{2b}\sum_{j=0}^{2b-1}\sum_{k=0}^{2b-1} a_j f(\theta_j,\phi_k)\overline{Y_\ell^m(\theta_j,\phi_k)} \tag{9}$$

where $b = 2^n$, $n = \lceil \log_2 2L \rceil$ and the grid is defined for $\theta_j = \pi j/2b$ and $\phi_k = \pi k/b$. a_j is the normalization factor to compensate the effect of denser sampling as we move towards poles ($\theta = 0,\pi$) from equator ($\theta = \pi/2$) and is dependent on $\theta_j = \pi j/2b$

$$a_j = \frac{2\sqrt{2}}{2b}\sin\left(\frac{\pi j}{2b}\right)\sum_{q=0}^{b-1}\frac{1}{2q+1}\sin\left([2q+1]\frac{\pi j}{2b}\right), \quad j = 0,\ldots,2b-1 \tag{10}$$

We discuss fast algorithms for computing spherical harmonic transform of a bandlimited signal in the next subsection.

C. Relevant Research Work

The spherical harmonic transform lacks a fast transform algorithm. The computational complexity of the direct computation of the transform algorithm is $O(L^3)$ [15]. Much research has been done to search for a fast and stable transform algorithm. In this subsection, we survey and compare some algorithms for the fast spherical harmonic transform.

Orszag [16] has described a fast evaluation scheme based on the low-order WKB approximation for Sturm-Liouville

eigenfunction transforms including the associated Legendre transform. His algorithm enables the computational complexity $O(L^2 \log L)$ for $m = 0$. However, for higher spherical harmonics order m, although his scheme is applicable, it is unlikely to be effective in both the precision and computational time [17]. Mohlenkamp [18] proposed a wavelet approach method which attains fast transform algorithms run in time $O(L^{5/2} \log L)$ and $O(L^2 \log^2 L)$.

Several researchers accelerate the spherical harmonic transform through the improvement of associated Legendre function. Alpert and Rokhlin [19] showed that the Chebyshev polynomial expansion can transform into associated Legendre polynomial expansion within $O(L^2)$ computation. Beylkin et al. [20] showed the same result using a wavelet approach. The L degree Chebyshev polynomial expansion can be evaluated in time $O(L \log L)$ using FFT. As L degree spherical harmonic transform requires L different Legendre transform, the overall computational complexity of spherical harmonic transform to $O(L^2 \log L)$. However, their approaches are based on the similarity of both the Legendre polynomials and the Chebyshev polynomials, so the computation will be slower of larger m. Apart from the methods mentioned above, Driscoll and Healy presented the precise, relatively efficient and most widely used spherical harmonic transform algorithm with complexity $O(L^2 \log^2 L)$, which takes into account the power of FFT and recurrence relation of Legendre polynomials. We provide the details of this algorithm in the next section.

In this work, we extend the method proposed presented by Driscoll and Healy to calculate the spherical harmonic transform of a bandlimited signal. We identify the components and independent elements of the method that can be implemented in parallel and present the modified algorithm so that the overall computational complexity can be reduced by running these identified parts in parallel. We use the Compute Unified Device Architecture (CUDA) to run the modified algorithm in parallel. The brief introduction of CUDA is given in next subsection.

D. CUDA and its Toolbox in Matlab - Jacket

CUDA is a parallel architecture developed by Nvidia and it enables parallel computation on GPUs. Because originally GPUs are designed to devote to data processing rather than data caching and flow control, the main difference between CPUs and GPUs is that GPUs have a parallel throughput architecture that performs relatively independent computations on large quantities of threads on different kernels in parallel and therefore are very suitable for computations of large size.

We use Matlab to simulate the computational time costs of parallel and sequential spherical harmonic transform algorithms. Jacket, developed by AccelerEyes, provides thousands of Matlab functions to run on GPUs and enables implementing parallel computation based on CUDA to run on Matlab. It accomplishes this by automatically wrapping the Matlab language into a GPU compatible form. By simply casting input data to Jacket's GPU data structure, MATLAB functions are transformed into GPU functions. Jacket also preserves the interpretive nature of the Matlab language by providing realtime, transparent access to the GPU compiler. It has been used by Tubbs and Tsai to accelerate lattice Boltzmann model for shallow water flow and mass transport [21].

III. Spherical Harmonic Transform in Parallel Computation

In this section, we revisit the most commonly used algorithm to calculate the spherical harmonic transform, proposed by Driscoll and Healy [7]. We then present the modified algorithm suitable to run on parallel architecture. Finally, we discuss the parallel implementation of their algorithm and compare the computational complexity.

A. Driscoll and Healy Algorithm

Driscoll and Healy used the classical definition of spherical harmonics (2) and exploited the power of FFT and the recurrence relation of associated Legendre polynomials to calculate the spherical harmonic transform for the bandlimited signal more efficiently. The algorithm employs the fast Fourier transform (FFT) to determine the contribution of signal component along ϕ component and uses the recurrence relation of associated Legendre transform to calculate the signal component which varies with θ.

The equiangular grid sampling on the sphere to separate the projection of signal on spherical harmonics along θ and ϕ. Using the spherical harmonics definition in (2), the transform representation in (9) can be expanded as

$$f_\ell^m = (-1)^m N_\ell^m \frac{\sqrt{2\pi}}{2b} \sum_{j=0}^{2b-1} a_j \times \Big(\sum_{k=0}^{2b-1} f(\theta_j, \phi_k) e^{-im\phi_k} \Big) P_\ell^m(\cos\theta_j) \tag{11}$$

we see that the inner sum is independent of θ component and is equivalent to the $2b$ length Fourier transform with respect to ϕ. By taking the FFT along ϕ_k for each θ_j (each row), $f(\theta_j, \phi_k)$ is transformed into $f(\theta_j, m)$ and the the inner summation become a function of θ_j and m and f_ℓ^m in (11) can be written as

$$f_\ell^m = (-1)^m N_\ell^m \frac{\sqrt{2\pi}}{2b} \sum_{j=0}^{2b-1} a_j f(\theta_j, m) P_\ell^m(\cos\theta_j) \tag{12}$$

and the problem is reduced to apply Legendre transform along θ to determine the spherical harmonic coefficients.

B. Parallel Implementation

The main idea of the above mentioned algorithm is to deal with the signal components along latitude θ and longitude ϕ separately by taking advantage of their different structure in (11). The FFT is used to determine the component along ϕ and the associated Legendre transform is applied to calculate the component along θ, which is efficiently computed by projecting the signal onto cosine basis using discrete cosine

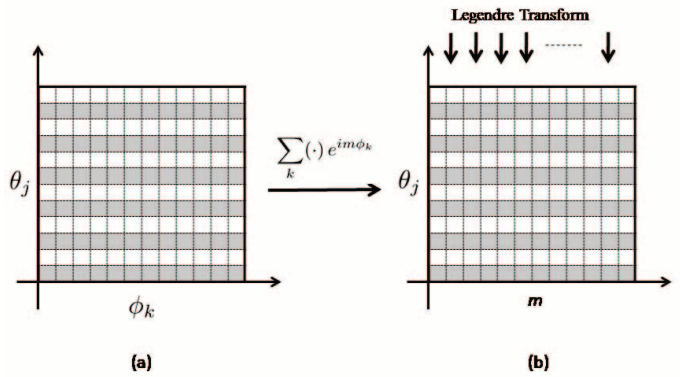

Fig. 1. (a)$f(\theta_j, \phi_k)$ is transformed into (b) $f(\theta_j, m)$ by taking FFT along ϕ_k for each θ_j. The Legendre transform is then applied along θ_j for each fixed m to compute the spherical harmonic coefficients.

transform (DCT) which gives the spherical harmonic coefficients f_m^m as $P_m^m(\cos\theta)$ is a multiple of$(\cos\theta)^m$ [7]. The recurrence relation is then employed to determine the spherical harmonic coefficients f_ℓ^m for $m < \ell \leq L$. The pictorial view of this algorithm is given in Fig. 1.

For the ease of parallel implementation, we divide the algorithm into two parts. First, we take the FFT for the longitude ϕ part of the given bandlimited signal $f(\theta_j, \phi_k)$ with size $2b \times 2b$ to make the result $f(\theta_j, m)$ suitable for the usage of the recurrence algorithm in the Legendre transform. We identify that we can distribute the $2b$ rows of the signal matrix of size $2b \times 2b$, which varies with the longitude ϕ in different kernels of the GPU.

Now, we have a signal matrix $f(\theta_j, m)$ which is a function of spherical harmonics order m and samples across latitude θ_j, where the spherical harmonics order m is of the order L. So, we need to employ the DCT and then the recurrence relation on each row of the signal matrix for each fixed m. Instead of running the same algorithm $O(L)$ times sequentially, we can distribute $f(\theta_j, m)$ for different spherical harmonics orders m into each kernel of GPU and use the recurrence calculations of different data with the same algorithm in each kernel. Again, we need $O(L)$ number of processors as compared to $O(L^2)$ number of processors which are needed in the algorithm given by [11]. The algorithm of the above process is presented in Algorithm 1. We provide the analysis of improvement in the computational complexity in the next subsection.

C. Computational Complexity

For a given bandlimited signal with maximum spherical harmonic degree L, we need the $2b \times 2b$ number of samples where b is of the order $O(L)$ as defined in Theorem 1. First, we consider the computational complexity of the sequential Driscoll and Healy algorithm to calculate the spherical harmonic coefficients, which is a two step process and composed of FFT followed by the Legendre transform as discussed in previous section. The computational time to determine

Algorithm 1 Parallel Algorithm of input $f(\theta_j, \phi_k)$

distribute $2b$ rows, varying along the longitude ϕ_k in different kernel of GPU
for EACH kernel of GPU **do**
  do FFT for the distributed row
end for
{We can get $f(\theta_j, m)$ after this}
distribute $2b$ columns, each with a fixed m, in different kernel of GPU
for EACH kernel of GPU **do**
  do DCT for the distributed column
  generate f_ℓ^m for all $m \leq \ell \leq L$ using fast Legendre transform of given in [7] for this fix m
end for

Legendre transform is $O(L^2\log^2 L)$ which is dominant over the time $O(L^2 \log L)$ to calculate the FFT, thus, the overall complexity of the sequential algorithm is $O(L^2\log^2 L)$ [7]. In our modified algorithm, we proposed to parallelize both the FFT and Legendre transform. Using the $O(L)$ number of processors, the complexity to compute FFT along ϕ direction reduces to $O(L\log L)$. Followed by FFT, we determine the Legendre transform as given in (12) for each order m in parallel to determine the spherical harmonic coefficients f_ℓ^m for all $\ell \leq m \leq L$. Since the m is of the order L, we again need $O(L)$ number of processors and the time taken by each processor would be $O(L\log^2 L)$ which is the overall computational complexity of the proposed algorithm.

IV. Simulation Results and Discussion

We compare the time complexity of conventional sequential algorithm [7] and proposed parallel algorithm to determine spherical harmonic transform for a given bandlimited signal. The architecture used for the experiments is a dual core with Intel Core i5 M 480 2.67 GHz, with a Nvidia GeForce GTX s480 with 1536 MB GDDR5 (480 cores in the GPU). The algorithms were run under Windows 7 64bits, MATLAB R2011a, Jacket 1.8 and CUDA 4.0. We expect to obtain the *similar* comparative results on other GPU's with a difference of shift of intersection point between performance curves of CPU and GPU. The signal under consideration is the topographic map of Mars with maximum spherical harmonic degree 791. We simulate the computational complexity for different bandwidths. For each bandwidth, we first truncate the signal spectrum up to that degree and determine the signal using spherical harmonics synthesis equation (6). Since there is a symmetry between f_ℓ^m and f_ℓ^{-m} given in (8), we only compute the spherical harmonic coefficients f_ℓ^m for $0 \leq m \leq l$.

Fig. 2 shows the computational time to compute all spherical harmonic coefficients f_ℓ^m with $0 \leq \ell \leq L$, $0 \leq m \leq l$ for a bandlimited signal with maximum spherical harmonic degree L. We present the simulation result for maximum spherical harmonics degree of 250. It is evident that the proposed parallel implementation of the algorithm using GPU

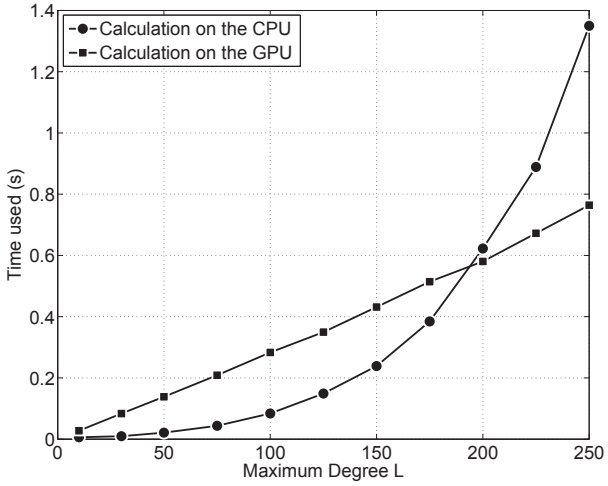

Fig. 2. Comparison of computational time taken by CPU and GPU to calculate the spherical harmonic coefficients f_ℓ^m for $0 \leq \ell \leq L$, $\ell \leq m \leq \ell$ for a given bandlimited signal with maximum spherical harmonic degree.

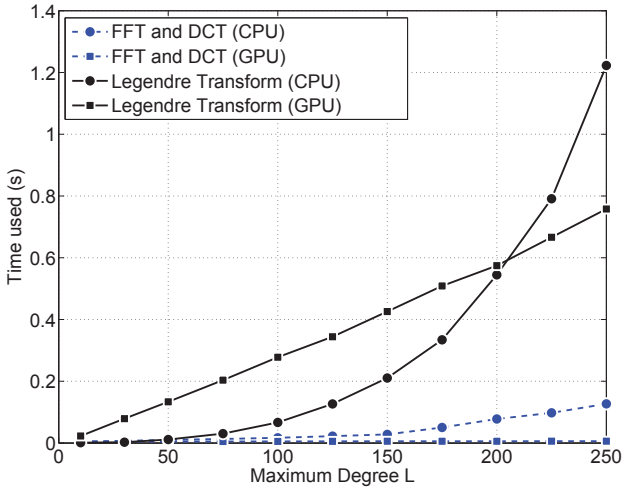

Fig. 3. Computational time taken by CPU and GPU to determine the Legendre transform and a combination of FFT and DCT.

performs efficiently than the sequential algorithm running on CPU for degree $L > 190$ and the trend is similar to theoretical asymptotic computational complexity. If we analyze the computational complexities of two steps involved in computations separately as shown in Fig. 3, we see that GPU performs better in computing FFT and DCT for $L > 35$, but inefficient in computing the Legendre transform for lower spherical harmonics degree which contributes significantly in the overall complexity. This is due to the fact that the time taken by GPU to perform one mathematical operation is more than the time taken by CPU, but the GPU has a power that it can run multiple instances in parallel and there are more number of operations $O(L\log^2 L)$ than the number of parallel instances L because of inherent recursion involved in Legendre transform.

The proposed parallel algorithm, although, asymptotically better, calculates the spherical harmonic coefficients more efficiently than the sequential algorithm for higher maximum spherical harmonic degree. This is because of the recursion involved in the algorithm that we can not further decouple the computation of fixed order different degree spherical harmonic coefficients.

V. Conclusions and Future Works

We considered and analyzed the parallel implementation of Driscoll and Healy spherical harmonic transform algorithm using CUDA. We extracted the independent operations in the existing method, presented the modified algorithm to calculate the spherical harmonic coefficients using parallel architecture for a given bandlimited signal with maximum spherical harmonics degree L and provided parallel implementation details using $O(L)$ number of processors. The theoretical analysis shows that the proposed parallel algorithm reduces the computational complexity to $O(L\log^2 L)$ as compared to the sequential algorithm complexity $O(L^2\log^2 L)$. This reduction in computational complexity has been also shown in simulation using CUDA.

The inherent coupling in the recursive definition of the associated Legendre polynomials puts the limitation on further parallelization of the algorithm. Parallel computation is quite suitable for the large data input and the non-recursive algorithm that determines each spherical harmonic coefficient independently. The future research direction is to explore and develop the suitable parallel algorithm to determine spherical harmonic transform that eradicates the recursion in Legendre transform so that each spherical harmonic coefficient can be computed independently and efficiently. In this way, we will need $O(L^2)$ number of processors and we expect that the computational complexity can be considerably reduced.

Acknowledgment

This work was supported by the Australian Research Council Discovery Grant DP1094350. We are thankful to Dr. Bradley Treeby for discussions on integration of CUDA with the Matlab and optimization procedures.

References

[1] F. J. Simons, F. A. Dahlen, and M. A. Wieczorek, "Spatiospectral concentration on a sphere," *SIAM Review*, vol. 48, no. 3, pp. 504–536, 2006.

[2] D. N. Spergel and et. al., "Wilkinson microwave anisotropy probe (WMAP) three year results: Implications for cosmology," *The Astrophysical Journal*, vol. 170, p. 377, Feb. 2007.

[3] C. Han, B. Sun, R. Ramamoorthi, and E. Grinspun, "Frequency domain normal map filtering," *ACM Transactions on Graphics*, vol. 26, no. 3, pp. 28:1–28:12, Jul. 2007.

[4] P. Yu, P. Grant, Y. Qi, X. Han, F. Segonne, R. Pienaar, E. Busa, J. Pacheco, N. Makris, R. Buckner, P. Golland, and B. Fischl, "Cortical surface shape analysis based on spherical wavelets," *IEEE Trans. Med. Imag.*, vol. 26, no. 4, pp. 582–597, Apr. 2007.

[5] B. T. T. Yeo, W. Ou, and P. Golland, "On the construction of invertible filter banks on the 2-sphere," *IEEE Trans. Image Process.*, vol. 17, no. 3, pp. 283–300, Mar. 2008.

[6] T. S. Pollock, T. D. Abhayapala, and R. A. Kennedy, "Introducing space into MIMO capacity calculations," *Journal on Telecommunications Systems*, vol. 24, pp. 415–436, Oct. 2003.

[7] J. R. Driscoll and D. M. Healy, "Computing fourier transforms and convolutions on the 2-sphere," *Advances in Applied Mathematics*, vol. 15, no. 2, pp. 202–250, Jun. 1994.
[8] W. J. can der Laan, "Accelerating wavelet lifting on graphics hardware using cuda," *Parallel and Distributed Systems*, vol. 22, pp. 132–146, Jan. 2011.
[9] S. Zhao and C. Zhou, "Accelerating spatial clustering detection of epidemic disease with graphics processing unit," in *2010 18th International Conference on Geoinformatics*, Jun. 2010, pp. 1–6.
[10] F. Feinbube, B. Rabe, M. von L andwis, and A. Polze, "Nqueens on cuda: Optimization issues," in *2010 Ninth International Symposium on Parallel and Distributed Computing (ISPDC)*, Jul. 2010, pp. 63–70.
[11] M. A. Inda, R. H. Bisseling, and D. K. Maslen, "On the efficient parallel computation of legendre transforms," *Mathematics Subject Classification*, no. 1, pp. 271–303, Jun. 2001.
[12] H. Xiao and Y. Lu, "Parallel computation for spherical harmonic synthesis and analysis," *Computers and Geosciences*, vol. 33, no. 3, pp. 311 – 317, Mar. 2007.
[13] J. J. Sakurai, *Modern Quantum Mechanics*, 2nd, Ed. Reading, MA: Addison Wesley, 1994.
[14] K. M. Górski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, "HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere," *The Astrophysical Journal*, vol. 622, pp. 759–771, Apr. 2005.
[15] R. Suda and M. Takami, "A fast spherical harmonics transform algorithm," *Mathematics of Computation*, vol. 71, no. 238, pp. 703–715, Apr. 2002.
[16] S. A. Orszag, "Fast eigenfunction transforms," *Academic Press: Science and Computers, Adv. Math. Supplementary Studies*, vol. 10, pp. 23–30, Oct. 1986.
[17] R. S. A. Mori and M. Sugihara, "An improvement on orszag's fast algorithm for legendre polynomial transform," *Trans. Inform. Process. Soc. Japan*, vol. 40, no. 9, pp. 3612–3615, Nov. 1999.
[18] M. J. Mohlenkamp, "A fast transform for spherical harmonics," Ph.D. dissertation, Yale University, 1997.
[19] B. K. Alpert and V. Rokhlin, "A fast algorithm for the evaluation of legendre expansions," *SIAM J. Sci. Stat. Comput.*, vol. 12, no. 7, pp. 158–179, Jan. 1991.
[20] G. Beylkin, R. R. Coifman, and V. Rokhlin, "Fast wavelet transforms and numerical algorithms i," *Comm. Pure Appl. Math*, vol. 44, pp. 141–183, Mar. 1991.
[21] K. R. Tubbs and F. T.-C. Tsai, "Gpu accelerated lattice boltzmann model for shallow water flow and mass transport," *International Journal for Numerical Method in Engineering*, vol. 86, pp. 316–334, Apr. 2011.