Universidade do Minho
Escola de Engenharia
Departamento de Informática

Jorge Francisco Teixeira Bastos da Mota

Study on FFT on the GPU

January 2022

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Jorge Francisco Teixeira Bastos da Mota

Study on FFT on the GPU

Master dissertation
Integrated Master's in Informatics Engineering

Dissertation supervised by
Supervisor
Co-supervisor (if any)

January 2022

ABSTRACT

Write abstract here (en) or import corresponding file

KEYWORDS keywords, here, comma, separated.

RESUMO

Escrever aqui resumo (pt) ou importar respectivo ficheiro

PALAVRAS-CHAVE palavras, chave, aqui, separadas, por, vírgulas

CONTENTS

1

INTRODUCTION

1.1 CONTEXTUALIZATION

Empty

1.2 MOTIVATION

Empty

1.3 OBJECTIVES

The main objective of this dissertation is to provide efficient FFT alternatives in GLSL compared with dedicated tools for high performance of FFT computations like NVIDIA cuFFT library, while analysing the intrinsic of a good Fast Fourier Transform implementation on the GPU. To accomplish the main objective there are two stages taken in consideration, "*Analysis of CUDA and GLSL kernels*" to be well settled in their differences and to have a reference for the second stage "Analysis of cuFFT and GLSL FFT" which will cluster the study's main objective.

To compose a final verdict conclusion, we will use as case of study applications with implementation of the FFT in the field of Computer Graphics that require realtime performance.

1.4 DOCUMENT ORGANIZATION

Empty

2

STATE OF THE ART

2.1 FOURIER TRANSFORM

2.1.1 *What is Fourier Transform*

The **Fourier Transform** is a mathematical method to transform the domain refered to as *time* of a function, to the *frequency* domain, intuitively the Inverse Fourier Transform is the corresponding method to reverse that process and reconstruct the original function from the one in *frequency* domain representation.

Although there are many forms, the Fourier Transform key definition can be described as:

$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-ift} dt \qquad \text{Forward Fourier Transform}$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(f) e^{-ift} df \qquad \text{Inverse Fourier Transform}$$

- $x(t), \forall t \in \mathbb{R} \rightarrow$ function in *time* domain representation with real t.
- $X(f), \forall f \in \mathbb{R} \rightarrow$ function in *frequency* domain representation with real f, also called the Fourier Transform of $x(t)$
- $i \rightarrow$ imaginary unit $i = \sqrt{-1}$

This formulation shows the usage of complex-valued domain since the imaginary unit i doesn't represent a value in the set of real numbers, making the fourier transform range from real to complex values, one complex coefficient per frequency $X : \mathbb{R} \rightarrow \mathbb{C}$

If we take into account the Euler's formula, we can replace the Fourier Transform for an equivalent, fragmenting the euler constant for a sine and cosine

$$e^{ix} = \cos x + i \sin x \qquad \text{Euler's Formula}$$

$$X(f) = \int_{-\infty}^{+\infty} x(t)(\cos(-ft) + i \sin(-ft)) dt \tag{1}$$

Hence, we can break the Fourier Transform apart into two formulas that give each coefficient of the sine and cosine components as functions without dealing with complex numbers.

$$
\begin{aligned}
X_a(f) &= \int_{-\infty}^{+\infty} x(t)\cos(ft)dt \\
X_b(f) &= \int_{-\infty}^{+\infty} x(t)\sin(ft)dt
\end{aligned}
\tag{2}
$$

The above definition of the Fourier IntegralForward Fourier Transform can only be valid if the integral exists for every value of the parameter f. This model of the fourier transform applied to infinite domain functions is called **Continous Fourier Transform** and its targeted to the calculation of the this transform directly to functions with only finite discontinuities in $x(t)$.

2.1.2 *Where it is used*

It's noticieable the presence of Fourier Transforms in a great variety of apparent unrelated fields of application, even the FFT is often called ubiquitous[1] due to its effective nature of solving a great hand of problems for the most intended complexity time. Some of the fields of application include Applied Mechanics, Signal Processing, Sonics and Acoustics, Biomedical Engineering, Instrumentation, Radar, Numerical Methods, Electromagnetics, Computer Graphics and more Brigham (1988).

One of the most well known cases of application is **Signal Analysis**, the Fourier Transform is probably the most important tool for analyzing signals, when representing a signal with amplitude as function of time, a signal can be translated to the frequency domain, a domain that consists of signals of sines and consines waves of varied frequencies, but to calculate the coefficients of those waves we need to use the Fourier Transform.

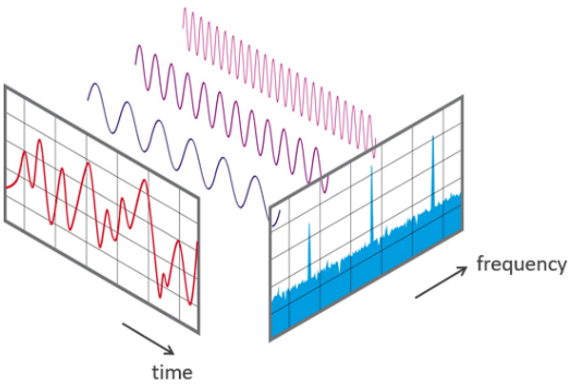

Figure 1: Time to frequency signal decomposition

1 present, appearing, or found everywhere.

Since the sines and consines waves are in simple waveforms they can then be manipulated with relative ease. This process is constantly present in communications since the transmission of data over wires and radio circuits through signals and most devices nowadays perform ir frequently

And much more applications such as polynomial multiplication Jia (2014), numerical integration, time-domain interpolation, x-ray diffracition ...

2.1.3 *Discrete Fourier Transform*

The Fourier Transform of a finite sequence of equally-spaced samples of a function is the called the **Discrete Fourier Transform** (DFT), it converts a finite set of values in *time* domain to *frequency* domain representation. Its the most important type of transform since it deals with a discrete amount of data and has the popular algorithm in which is the center of attention of fourier transforms, which can be implemented in machines and be computed by specialized hardware.

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N}kn} \qquad \text{Forward Discrete Fourier Transform}$$

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{\frac{i2\pi}{N}kn} \qquad \text{Inverse Discrete Fourier Transform}$$

Notably, the discrete version of the Fourier Transform has some obvious differences since it deals with a discrete time sequence, the first difference is the sum covers all elements of the input values instead of integrating the infinite domain of the function, but we can also notice that the exponential, similar to the aforesaid, divides the values by N (N being the total number of elements in the sequence) due to the inability to look at frequency and time ft continuously we instead take the k'th frequency over n.

We can have a more simplified expansion of this formula with:

$$X_k = x_0 + x_1 e^{\frac{i2\pi}{N}k} + ... + x_{N-1} e^{\frac{i2\pi}{N}k(N-1)}$$

Having this sum simplified we then only need to resolve the complex exponential, and we can do that by replacing the $e^{\frac{i2\pi}{N}kn}$ by the euler formula as mentioned before to reduce the maths to a simple summation of real and imaginary numbers.

$$X_k = x_0 + x_1(\cos b_1 + i \sin b_1) + ... + x_{N-1}(\cos b_{N-1} + i \sin b_{N-1}) \tag{3}$$

$$\text{where } b_n = \frac{2\pi}{N}kn$$

Finally we'll be left with the result as a complex number

$$X_k = A_k + iB_k$$

EXAMPLE Let us now follow an example of calculation of the DFT for a sequence x with N number of elements.

$$x = \begin{bmatrix} 1 & 0.707 & 0 & -0.707 & -1 & -0.707 & 0 & 0.707 \end{bmatrix}$$
$$N = 8$$

With this sequence we now want to transform it into the frequency domain, and for that we need to apply the Discrete Fourier Transform to each element $x_n \rightarrow X_k$, thus, for each k'th element of X we apply the DFT for every element of x.

$$X_0 = 1 \cdot e^{-\frac{i2\pi}{8} \cdot 0 \cdot 0} + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 0 \cdot 1} + ... + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 0 \cdot 7}$$
$$= (0 + 0i)$$
$$X_1 = 1 \cdot e^{-\frac{i2\pi}{8} \cdot 1 \cdot 0} + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 1 \cdot 1} + ... + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 1 \cdot 7}$$
$$= (4 + 0i)$$

...

$$X_7 = 1 \cdot e^{-\frac{i2\pi}{8} \cdot 7 \cdot 0} + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 7 \cdot 1} + ... + 0.707 \cdot e^{-\frac{i2\pi}{8} \cdot 7 \cdot 7}$$
$$= (4 + 0i)$$

And that will produce our complex-valued output in frequency domain, as simple as that.

$$X = \begin{bmatrix} 0i & 4 + 0i & 0i & 0i & 0i & 0i & 0i & 4 + 0i \end{bmatrix}$$

Matrix multiplication

The example shown above is done sequentially as if each frequency pin is computed individually, but there's a way to calculate the same result by using matrix multiplication Rao and Yip (2018). Since the operations are done equally without any extra step we can group all analysing function sinusoids ($e^{-\frac{i2\pi}{N}kn}$),

$$W = \begin{bmatrix} \omega^{0\cdot 0} & \omega^{1\cdot 0} & \dots & \omega^{(N-1)\cdot 0} \\ \omega^{0\cdot 1} & \omega^{1\cdot 1} & \dots & \omega^{(N-1)\cdot 1} \\ \vdots & \vdots & \ddots & \vdots \\ \omega^{0\cdot (N-1)} & \omega^{1\cdot (N-1)} & \dots & \omega^{(N-1)\cdot (N-1)} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & \omega & \dots & \omega^{(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \dots & \omega^{(N-1)\cdot (N-1)} \end{bmatrix}$$

where $\omega = e^{-\frac{i2\pi}{N}}$

The substitution variable ω allows us to avoid writing extensive exponents.

The symbol W represents the transformation matrix of the Discrete Fourier Transform, also called DFT matrix, and its inverse can be defined as.

$$W^{-1} = \frac{1}{N} \cdot \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & \omega & \dots & \omega^{(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \dots & \omega^{(N-1)\cdot(N-1)} \end{bmatrix}$$

$$\text{where } \omega = e^{\frac{i2\pi}{N}}$$

By using this matrix multiplication form we can have a more efficient way to compute the DFT in hardware.

$$X = W \cdot x \qquad \text{Matrix DFT}$$

$$x = W^{-1} \cdot X \qquad \text{Matrix IDFT}$$

Moreover we might also want to normalize the matrix by $\sqrt{N}$ for both Matrix DFT and IDFT instead of just normalizing the IDFT by N, that will make W a unitary matrix Horn and Johnson (2012). The advantage of using a unitary matrix is that we only need to reasign the constant substution variable ω to be able to invert the dft, the matrix multiplication stays the same for both DFT and IDFT. Nevertheless later we will verify that the use of sqrt function isn't desirable for the implementation of any dft.

EXAMPLE Continuing the example 2.1.3, we can adapt the aplication of the DFT to the matrix multiplication form.

$$W = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & \omega & \dots & \omega^{7} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{7} & \dots & \omega^{49} \end{bmatrix}$$

$$\text{where } \omega = e^{\frac{i2\pi}{8}}$$

$$X = W \cdot x = W \cdot \begin{bmatrix} 1 \\ 0.707 \\ \vdots \\ 0.707 \end{bmatrix} = \begin{bmatrix} 0 \\ 4+0i \\ \vdots \\ 4+0i \end{bmatrix}$$

It's conspicuous that the complexity time for each multiplication of every singular term of the sequence with the complex exponential value is $O(N^2)$, hence, the computation of the Discrete Fourier Transform rises exponentially as we use longer sequences. Therefore, over time new algorithms and techniques where developed to increase the performance of this transform due to its usefulness.

2.2 FAST FOURIER TRANSFORM

Empty

2.2.1 *Computation of FFT*

Empty

2.3 RELATED WORK

Empty

BIBLIOGRAPHY

E Oran Brigham. *The fast Fourier transform and its applications*. Prentice-Hall, Inc., 1988.

Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

Yan-Bin Jia. Polynomial multiplication and fast fourier transform. *Com S*, 477:577, 2014.

Kamisetty Ramam Rao and Patrick C Yip. *The transform and data compression handbook*. CRC press, 2018.

NB: place here information about funding, FCT project, etc in which the work is framed. Leave empty otherwise.