Applied Ocean Research 74 (2018) 227–236

Contents lists available at ScienceDirect

Applied Ocean Research

journal homepage: www.elsevier.com/locate/apor

Comparison study between the Fourier and the Hartley transforms for the real-time simulation of the sea surface elevation

J.M. Varela [a], German Rodriguez [b], C. Guedes Soares [a,*]

[a] Centre for Marine Technology and Ocean Engineering (CENTEC), Instituto Superior Técnico, Universidade de Lisboa, Av. Rovisco Pais, 1049-001, Lisboa, Portugal

[b] Applied Marine Physics and Remote Sensing Group, Institute of Environmental Studies and Natural Resources (iUNAT), and Physics Department, Universidad de Las Palmas de G. C., 35017, Las Palmas, Spain

ARTICLE INFO

Article history:
Received 25 July 2017
Received in revised form 17 February 2018
Accepted 2 March 2018

Keywords:
Hartley transform
Fourier transform
Sea surface elevation
Numerical simulation
Signal processing
Algorithms

ABSTRACT

The paper presents the results of a comparison study between the Fourier and the Hartley transforms for the real-time simulation of the sea surface elevation in 3D. Fourier transforms are currently the most used and efficient method for obtaining realistic ocean scenes in interactive Virtual Environments. Although the Fast Fourier Transform has been the preferred choice for this type of simulations, the study reveals that the Fast Hartley Transform can be a valid alternative, and even have some advantages compared to the former. The study mainly focuses on the performance and memory aspects, which are decisive factors for real-time applications with ocean scenes, such as ship bridge simulators. The methodology to obtain the sea surface elevation in time domain from a sea state defined by a directional wave spectrum is also described.

1. Introduction

Integral transforms are commonly used in digital signal processing for converting wave field representations from the time-space to the frequency-wavenumber domains, and vice-versa, by applying the corresponding inverse transforms. Ocean wave fields are a typical case where this duality of representations is of most importance. In particular, the sea surface elevation in time-space domain can be obtained by transforming the directional wave spectrum, which defines the sea state in frequency domain.

A review of the literature reveals that the Fourier Transform (FT), or more specifically the Fast Fourier Transform (FFT) algorithm, developed by Cooley and Tukey [1] in 1965, is still the most used transform to simulate numerically and in real-time[1] the sea surface elevation of a sea state represented by a directional wave spectrum. However, the Hartley Transform (HT) developed by Hartley [2] in 1942, and its highly optimized algorithm, the Fast Hartley Transform (FHT) presented by Bracewell [3] in 1984, is advocated by some researchers as a valid alternative to the FFT, which may even have better performance and consume less computational resources. For applications that require the representation of the sea surface in real-time, such as Ship Bridge Simulators (SBS), the time spent in the calculation is of major importance, and therefore a comparison study to ascertain which one of the mentioned transforms has better performance in sea surface simulation, is justified.

The discussion about the advantages of the FHT over the FFT is more than 30 years old. Bracewell [3,4] argued that, for specific cases, the FHT preforms faster for spectral analysis and convolution, because it requires only real arithmetic operations, contrarily to the FFT, which requires complex arithmetic. Additionally, the inverse FHT is identical to its direct transform and therefore the same algorithm can be used for the analysis and synthesis of the signals. In 1985, Bold [5] showed that the FHT of a real sequence could be computed at most only 2 times faster than the same sequence using a complex FFT. However, he also alleged that sophisticated FFT algorithms could achieve the same speedup factor. The similarities of the FHT and the FFT in their simplest form were also highlighted. In fact, the complex FT of a real function and its HT can be expressed in terms of each other (Buneman [6]). Consequently, any FFT can thus be converted into a FHT, by only a few indexing changes. Later in the same year, Bracewell [7] expanded the one-dimensional FHT to bidimensional signals by direct analogy with the two-dimensional FFT algorithms. For a two-dimension array of data, the method allows to derive the two-dimensional HT by computing the one-dimensional FHT of the rows one by one, and then transforming

* Corresponding author.
E-mail address: c.guedes.soares@centec.tecnico.ulisboa.pt (C. Guedes Soares).

[1] Within the context of the current paper, real-time means a simulation running at a minimum speed of 30 cycles per second. This value is commonly accepted by the research community as a reference for an acceptable interaction between the user and the scenario.

https://doi.org/10.1016/j.apor.2018.03.002

the columns. In the subsequent years, FHT algorithms were developed by Hau and Bracewell [8,9] in which a one-dimensional FHT is used to obtain a three-dimensional FFT. However, Sorensen [10] shows that the discussion about the benefits and disadvantages of using the FHT was not consensual at the time. Research work in this field continued with search and development of new optimized and faster algorithms to compute the DHT. In 1992, Meher [11] proposed a new algorithm, which he claimed to be faster than the previous ones developed by Bracewell [7] and by Boussakta and Holt [12]. Two years later, Unyal [13] analyses the efficiency of the computation of FFT real-valued data with respect to their operation counts. He concludes that the FHT algorithm computes faster than the real-valued FFT. The same conclusion is taken by Scott [14] in 2000, which states that using the FHT to compute the FFT is faster than computing the FFT directly.

So far, the FFT has been the primary choice of Computer Graphics (CG) developers to simulate the sea surface in interactive Virtual Environments (VE) and, as far as to the authors' knowledge, the FHT has not been properly explored in this field. Some significant research works developed by Mastin [15], Tessendorf [16], Fréchot [17], or Varela and Guedes Soares [18], all apply the FFT to compute the surface elevation in real-time. Moreover, Darles et al. [19] presents a survey of sea surface simulation techniques in CG, where this tendency is also confirmed. In fact, only a few research works such as Rodriguez [20], present an analysis and simulation of sea wave records using the FHT as an efficient real-valued alternative to the complex FFT. Although his work was limited to 1D simulations and the graphical representation of the sea surface was not a concern, it is a reference mark for the application of the FHT to the simulation of ocean waves in random seas.

The current paper presents a comparison study between the FFT and the FHT applied to the numerical simulation of the sea surface elevation. Comparison factors taken into consideration are the performance, measured by the time spent on the transform calculation each cycle, and the amount of memory required to execute the transform calculations. Additionally, some comments are added regarding implementation issues of both transforms.

The main contribution of the paper is to identify which of the mathematical transforms is more adequate to simulate the sea surface elevation in real-time in a 3D Virtual Environment, concerning the mentioned comparison factors. For this purpose, both algorithms are implemented in the real-time simulator developed by Varela and Guedes Soares [21].

The core of the paper starts in Section 2 with the theoretical background behind the Fourier and the Hartley Transforms and their optimized algorithms of the FFT and FHT. Section 3 describes the methodology used within the current study to simulate and visualise irregular seas in interactive VEs. The general approach to obtain surface images from wave-spectrum density functions is presented. The sea surface simulations are described in Section 4, which includes the software and hardware used in simulations, the experimental conditions and the cases studied. Implementation aspects regarding the input data setup and the core algorithm of mathematical transforms are presented in Section 5. The results are discussed in Section 6 and conclusions of the study are presented in Section 7.

2. Theoretical background

2.1. Fourier and Hartley Transforms

The Fourier transform of a continuous function, $\eta(t)$, of a continuous variable, t, is defined by:

$$F(f) = \int_{-\infty}^{\infty} \eta(t) e^{-i2\pi ft} dt \tag{1}$$

where f is also a continuous variable. The inverse Fourier transform can be written as:

$$\eta(t) = \int_{-\infty}^{\infty} F(f) e^{i2\pi ft} df \tag{2}$$

where the kernel transform function is:

$$exp(\pm 2\pi ft) = \cos(2\pi ft) \pm i \ \sin(2\pi ft) \tag{3}$$

Usually, t represents the time in seconds and f the frequency in Hertz. Nevertheless, the transforms can work with other variables and units. (Thus, for instance, if η is a function of the spatial position, F will be a function of wavenumber). Furthermore, in most practical cases, $\eta(t)$ is obtained by sampling a signal at evenly spaced intervals over a finite range of t and, consequently, the direct and inverse Fourier transforms must be evaluated by means of their discrete versions. That is, the discrete Fourier transform (DFT), given by:

$$F(k) = \sum_{n=0}^{N-1} \eta(n) e^{-i2\pi kn/N} \tag{4}$$

where $\eta(n)$ is a data sequence sampled from $\eta(t)$ and can be recovered by using the inverse discrete Fourier transform (IDFT), which can be written as:

$$\eta(n) = \frac{1}{N} \sum_{k=0}^{N-1} F(k) e^{i2\pi kn/N} \tag{5}$$

It is interesting to note that if η(t) is a real function, its Fourier transform is, in general, complex.

The Hartley transform (Hartley [2]) maps a time, or space, real-valued function into a real-valued frequency, or wavenumber, function. It can be expressed as:

$$H(f) = \int_{-\infty}^{\infty} \eta(t) cas(2\pi ft) dt \tag{6}$$

where the kernel transformation function, known as *cas* function is defined as:

$$\text{cas}(2\pi ft) = \cos(2\pi ft) + \sin(2\pi ft) \tag{7}$$

and the inverse Hartley transform is given by:

$$\eta(t) = \int_{-\infty}^{\infty} H(f) cas(2\pi ft) df \tag{8}$$

As in the case of the Fourier transform, the evaluation of the above integral transforms for sampled finite signals requires the use of discrete approximations. The discrete Hartley transform (DHT) is given by:

$$H(k) = \sum_{n=0}^{N-1} \eta(n) cas(\frac{2\pi kn}{N}) \tag{9}$$

and the corresponding inverse discrete Hartley transform (IDHT) can be written as:

$$\eta(n) = \sum_{k=0}^{N-1} H(k) cas(\frac{2\pi kn}{N}) \tag{10}$$

2.2. Fast Fourier and Hartley transform algorithms

The direct implementation of Eqs. (4) and (5) for a time series of N sample points is not an efficient procedure because it requires about N^2 arithmetic operations. That is why several fast algorithms, known as FFT algorithms, have been developed for the efficient computation of the DFT (see, e.g., Chu [22]). The vast