A Cooley–Tukey Algorithm for the Slant Transform

N. AHMED† and
M. C. CHEN‡

A Cooley–Tukey algorithm for the slant transform and its inverse is developed. It is shown that using this algorithm, one can compute the slant transform by means of a simple modification of the same Cooley–Tukey algorithm which is used to compute the Hadamard ordered Walsh–Hadamard transform.

KEY WORDS AND PHRASES: Cooley–Tukey algorithm, slant transform, Hadamard ordered Walsh–Hadamard transform.

CATEGORIES: 3.20, 3.24, 5.12.

1. INTRODUCTION

The notion of an orthogonal transform consisting of "slant" basis vectors was introduced by Enomoto and Shibata [1]. The term slant vector implies a sawtooth waveform decreasing in uniform steps over its length, and hence suitable for representing gradual brightness changes in an image line.

The work of Enomoto and Shibata was restricted to slant vectors of lengths 4 and 8. A generalization by Pratt, Welch, and Chen [2] led to the definition of the slant transform (ST). Since its development, the ST has been used successfully in the area of image processing, as evident from a recent paper by Pratt [3].

One of the factors that has contributed to the image processing applications of the ST is that it possesses a fast computational algorithm [3]. The signal flow graph associated with this algorithm for the case \( N = 8 \) is shown in Figure 1. It is clear that this algorithm cannot be identified as belonging to any specific class of existing algorithms. This leads one to conclude that the ST computational problem is essentially unrelated to the problems of computing other orthogonal transforms that are also used for image processing—e.g.

†Department of Electrical Engineering, Kansas State University, Manhattan, Kansas 66506, U.S.A.; formerly with Department of Computer Science also.
‡Philco Ford, 1100 NASA Road 1, Houston, Texas 77058; formerly with the Department of Computer Science, Kansas State University, Manhattan, Kansas 66506, U.S.A.
discrete Fourier transform (DFT) [4], Walsh–Hadamard transform (WHT) [4], discrete cosine transform (DCT) [5, 6], and Haar transform [4]. However, in this paper we show that this is not the case, since the ST can also be computed using an algorithm which belongs to the well-known class of Cooley–Tukey algorithms. Further it is shown that this new algorithm can be implemented by means of a simple modification of the algorithm used to compute the Hadamard ordered Walsh–Hadamard transform† (WHT)$_h$ [4], followed by a straight-forward shuffling process which results in a rearrangement of the ST coefficients in sequency ordered form.

2. MOTIVATION

There are two factors that motivate one to relate the ST to the Walsh–Hadamard transform. To consider the first of these, we examine Figure 2 which shows a comparison of the basis vectors for the ST and the Walsh ordered Walsh–Hadamard transform (WHT)$_w$. With respect to Figure 2, we make the following observations: (i) there is a strong resemblance between several basis vectors of the two transforms, and (ii) many of the mid-sequency vectors for the two transforms are identical. Next, we compare the definition of the ST with that of the (WHT)$_w$.

†Also known as the BIFORE transform [7].
If \( X(m), m=0, 1, \ldots, (N-1) \) denotes the given data sequence, then its ST is defined as

\[
\{L_N\} = \{S_N\}\{X_N\}
\]

(1)

where

\( \{L_N\} \) is the \((N \times 1)\) slant transform vector

\( \{X_N\} \) is the \((N \times 1)\) vector representation of the data sequence, and,

\( \{S_N\} \) is the \((N \times N)\) slant matrix, where \( N \) is an integer power of 2.

The matrix \( \{S_N\} \) in (1) possesses the following recursive definition [2, 3]:

\[
S_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

and

\[
S_N = \frac{1}{\sqrt{2}} \begin{bmatrix}
1 & 0 & 0 & i \\
0 & b_N & 0 & 0 \\
0 & 0 & I_{(N/2-1)} & 0 \\
0 & a_N & -b_N & 0
\end{bmatrix}
\begin{bmatrix}
I_{N/2} & I_{N/2} & \cdots & \cdots \\
0 & 0 & \cdots & \cdots \\
0 & 0 & \cdots & \cdots \\
0 & 0 & \cdots & \cdots
\end{bmatrix}
\begin{bmatrix}
S_{N/2} & 0 \\
0 & -S_{N/2}
\end{bmatrix}
\]

(2)
In (2), $I_{N/2-2}$ is the identity matrix of dimension $(N/2) - 2$, and the constants $a_n$, $b_n$ can be computed using the recursive relations

$$a_n = \frac{3N^2}{4N^2 - 1} \quad b_n = \frac{N^2 - 1}{4N^2 - 1}$$

where $\{W_n\}$ is the $(N \times 1)$ transform vector and $[H_n]$ is the $(N \times N)$ Hadamard matrix which can be generated recursively using the relation

$$[H_n] = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

and

$$[H_n] = \frac{1}{\sqrt{2}} \begin{bmatrix} I_{N/2} & I_{N/2} \\ I_{N/2} & -I_{N/2} \end{bmatrix} \begin{bmatrix} H_{N/2} & 0 \\ 0 & H_{N/2} \end{bmatrix}$$

Comparing (2) and (5) we observe that $S_n$ has one additional matrix factor—i.e., the first matrix to the left. However, each of the remaining matrix factors of $S_n$ has a structure which is identical to its counterpart in $H_n$. Hence we conclude that with some modification, it should be possible to compute the ST in (1) using an existing algorithm which is used to compute the (WHT)$_n$ in (4).

3. ST COOLEY-TUKEY ALGORITHM

It is known that the (WHT)$_n$ can be computed using a Cooley-Tukey type algorithm, as illustrated in Figure 3.

For the purpose of discussion, we consider the case $N = 8$. Then, from the matrix factorization indicated in (2), it follows that the ST coefficients $L(k), k = 0, 1, 2, ..., 7$ can be computed as shown in Figure 4. Clearly, the signal flow graph in Figure 4 can be obtained via a simple modification of that in Figure 3. Such modifications are essentially additional arithmetic operations which are indicated in Figure 4 by means of broken line boxes which enclose them. By virtue of the recursive relation in (2), it follows that these additional operations
C-T ALGORITHM FOR THE SLANT TRANSFORM

FIGURE 3 (WHT), Cooley-Tukey algorithm, \( N = 8 \).

FIGURE 4 ST algorithm, \( N = 8 \).
can be incorporated in the (WHT)k signal flow graph for any N of the form \(N = 2^n\).

The ST coefficients \(L(k), k = 0, 1, \ldots, (N - 1)\) are not in sequency ordered form. For example, examination of the sign changes of \(S_k\) shows that the sequencies of the coefficients \(L(k), k = 0, 1, \ldots, 7\) are 0,1,2,4,1,2,3, and 3. In what follows, we show that a straight-forward shuffling process enables one to rearrange the \(L(k)\) in a sequency ordered form.

Let \(L(p), p = 0, 1, \ldots, (N - 1)\) denote the ST coefficients in sequency ordered form and \(\gamma_p, p = 0, 1, \ldots, (N - 1)\) denote the index sequence associated with \(L(\gamma_p)\). The value of \(\gamma_p\) corresponding to \(p\) for a given \(N\) can be generated recursively starting with \(N = 2\). This is best illustrated by the diagram in Figure 5 for values of \(N = 2, 4, 8, 16,\) and 32. For example, using this diagram we obtain the following values for the indices \(\gamma_p\) corresponding to \(p = 0, 1, \ldots, (N - 1)\).

\[
\begin{align*}
N = 2, & \quad \begin{cases} p : & 0 \ 1 \\ \gamma_p : & 0 \ 1 \end{cases} \\
N = 4, & \quad \begin{cases} p : & 0 \ 1 \ 2 \ 3 \\ \gamma_p : & 0 \ 1 \ 2 \ 3 \end{cases} \\
N = 8, & \quad \begin{cases} p : & 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \\ \gamma_p : & 0 \ 1 \ 4 \ 5 \ 2 \ 6 \ 7 \ 3 \end{cases} \\
N = 16, & \quad \begin{cases} p : & 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \\ \gamma_p : & 0 \ 1 \ 8 \ 9 \ 4 \ 12 \ 13 \ 5 \end{cases} \\
N = 32, & \quad \begin{cases} p : & 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \\ \gamma_p : & 0 \ 1 \ 8 \ 9 \ 4 \ 12 \ 13 \ 5 \end{cases}
\end{align*}
\]

For example, use of the above information related to \(N = 8\) results in the final shuffling shown in Figure 4. The procedure illustrated in Figure 5 can easily be extended to higher values of \(N\) of the form \(N = 2^n\). In the present study, the above shuffling procedure has been verified for values of \(N\) up to 512.†

A corresponding algorithm can also be developed for the inverse slant transform (IST) which is defined as [see (1)]

\[
\{X_N\} = \{S_N\}^t \{L_N\}
\]  

†A listing of the related computer program is available [8].
where \( \{S_n\}^T \) denotes the transpose of \([S_n]\). The corresponding reshuffling needed to obtain the ST coefficients \( L(k) \) from the sequency ordered coefficients \( \hat{L}(k) \), \( k = 0, 1, \ldots, (N-1) \) can also be developed.

4. IMPLEMENTATION CONSIDERATIONS

The algorithms for the ST and its inverse have been implemented using FORTRAN as the programming language. The related computer programs are available to the interested reader [8].

5. CONCLUSIONS

It has been shown that the problem of computing the ST is closely related to that of computing other digital orthogonal transforms via Cooley-Tukey
algorithms. Further, the ST algorithm presented in this paper is realized by means of a simple modification of a well-known Cooley–Tukey algorithm used to compute a Walsh–Hadamard transform.

In general, more additions are required in the proposed algorithm relative to the available method. For example, the new algorithm requires four more additions for the case $N=8$. However, this minor disadvantage is compensated by other advantages inherent in the Cooley–Tukey structure as far as software implementation is concerned—e.g. “inplace” property, programming ease, and pedagogical considerations. In closing, we remark that it would be worthwhile for one to compare these two algorithms on the basis of execution times for various values of $N$. A listing of the computer program used to compute the $ST$ via the existing algorithm is available in [9].

References