GENERALIZED S TRANSFORM

Michael D. Adams and Faouzi Kossentini

Dept. of Elec. and Comp. Eng., University of British Columbia, 2356 Main Mall, Vancouver, BC, Canada V6T 1Z4
mdadams@ieee.org and faouzi@ece.ubc.ca

ABSTRACT

The generalized S transform (GST), a family of reversible integer-to-integer transforms inspired by the S transform, is proposed. This family of transforms is then studied in some detail. For example, the relationship between the GST and lifting scheme is discussed, and the effects of choosing different GST parameters are examined. Some examples of specific transforms in the GST family are also given.

1. INTRODUCTION

Reversible integer-to-integer transforms have become a popular tool for use in signal coding applications requiring lossless signal reproduction [1–6]. One of the best known transforms of this type is the S transform [1–4]. In this paper, we propose the generalized S transform (GST), a family of reversible integer-to-integer transforms based on the key ideas behind the S transform. We then study the GST in some detail. This leads to a number of interesting insights about transforms belonging to the GST family, including the S transform amongst others.

2. NOTATION AND TERMINOLOGY

Before proceeding further, a short digression concerning the notation and terminology used in this paper is appropriate. The symbols $\mathbb{Z}$ and $\mathbb{R}$ denote the sets of integer and real numbers, respectively. Matrix and vector quantities are indicated using bold type.

The symbol $\mathbf{I}_N$ is used to denote the $N \times N$ identity matrix. In cases where the size of the identity matrix is clear from the context, the subscript N may be omitted. A matrix $\mathbf{A}$ is said to be unimodular if $|\det \mathbf{A}| = 1$. The (i, j)th minor of the $N \times N$ matrix $\mathbf{A}$, denoted $\operatorname{minor}(\mathbf{A}, i, j)$, is the $(N-1)\times(N-1)$ matrix formed by removing the ith row and jth column from $\mathbf{A}$.

For $\alpha \in \mathbb{R}$, the notation $\lfloor \alpha \rfloor$ denotes the largest integer not more than α (i.e., the floor function), and the notation $\lceil \alpha \rceil$ denotes the smallest integer not less than α (i.e., the ceiling function). The symbol $\mathcal{Q}$ is used to denote a rounding operator. Such operators are defined only in terms of a single scalar operand. As a notational convenience, however, we use an expression of the form $\mathcal{Q}(\mathbf{x})$, where $\mathbf{x}$ is a vector/matrix quantity, to denote a vector/matrix for which each element has had the operator $\mathcal{Q}$ applied to it. In this paper, all rounding operators are assumed to satisfy the identity

$$\mathcal{Q}(x) = x \quad \text{for all } x \in \mathbb{Z}.$$

3. S TRANSFORM

One of the simplest and most ubiquitous reversible integer-to-integer mappings is the S transform [1–4], a nonlinear approximation to a particular normalization of the Walsh-Hadamard transform. The S transform is also well known as the basic building block for a reversible integer-to-integer version of the Haar wavelet transform [3, 4]. The forward S transform maps the integer vector $\begin{bmatrix} x_0 & x_1 \end{bmatrix}^T$ to the integer vector $\begin{bmatrix} y_0 & y_1 \end{bmatrix}^T$, and is defined most frequently (e.g., [3, 4]) as

$$\begin{bmatrix} y_0 \\ y_1 \end{bmatrix} = \begin{bmatrix} \lfloor \frac{1}{2}(x_0+x_1) \rfloor \\ x_0 - x_1 \end{bmatrix}.$$

This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The corresponding inverse transform is given (e.g., in [3]) by

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} t \\ t-y_1 \end{bmatrix}, \text{ where } t \stackrel{\triangle}{=} y_0 + \lfloor \tfrac{1}{2}(y_1+1) \rfloor. \tag{1}$$

While the S transform can be viewed as exploiting the redundancy between the sum and difference of two integers, namely that both quantities have the same parity (i.e., evenness/oddness), this overlooks a much more fundamental idea upon which the S transform is based. That is, as noted by Calderbank et al. [2], the S transform relies, at least in part, on lifting-based techniques.

4. GENERALIZED S TRANSFORM

By examining the S transform in the context of the lifting scheme, we are inspired to propose a natural extension to this transform, which we call the generalized S transform (GST). For convenience in what follows, let us define two integer vectors $\mathbf{x}$ and $\mathbf{y}$ as

$$\mathbf{x} \stackrel{\triangle}{=} \begin{bmatrix} x_0 & x_1 & \cdots & x_{N-1} \end{bmatrix}^T, \quad \mathbf{y} \stackrel{\triangle}{=} \begin{bmatrix} y_0 & y_1 & \cdots & y_{N-1} \end{bmatrix}^T.$$

The forward GST is a mapping from $\mathbf{x}$ to $\mathbf{y}$ of the form

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{C}\mathbf{x}\right), \tag{2}$$

where $\mathbf{B}$ is real matrix of the form

$$\mathbf{B} \stackrel{\triangle}{=} \begin{bmatrix} 1 & b_1 & b_2 & \cdots & b_{N-1} \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix},$$

$\mathbf{C}$ is a unimodular integer matrix defined as

$$\mathbf{C} \stackrel{\triangle}{=} \begin{bmatrix} c_{0,0} & c_{0,1} & c_{0,2} & \cdots & c_{0,N-1} \\ c_{1,0} & c_{1,1} & c_{1,2} & \cdots & c_{1,N-1} \\ c_{2,0} & c_{2,1} & c_{2,2} & \cdots & c_{2,N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{N-1,0} & c_{N-1,1} & c_{N-1,2} & \cdots & c_{N-1,N-1} \end{bmatrix},$$

and $\mathcal{Q}$ is a rounding operator. By examining (2), one can easily see that this transform maps integers to integers. In the absence of the rounding operator $\mathcal{Q}$, the GST simply degenerates into a linear transform with transfer matrix $\mathbf{A}$, where $\mathbf{A} \stackrel{\triangle}{=} \mathbf{B}\mathbf{C}$. Thus, the GST can be viewed as a reversible integer-to-integer mapping that approximates the linear transform characterized by matrix $\mathbf{A}$. The inverse GST is given by

$$\mathbf{x} = \mathbf{C}^{-1}\left(\mathbf{y} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right)\right). \tag{3}$$

Note that $\mathbf{C}^{-1}$ is an integer matrix, since by assumption $\mathbf{C}$ is a unimodular integer matrix. To show that (3) is, in fact, the inverse of (2), one need only observe that due to the form of $\mathbf{B}$, for any two $N \times 1$ vectors $\mathbf{u}$ and $\mathbf{v}$:

$$\mathbf{v} = \mathbf{u} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{u}\right) \tag{4}$$

implies

$$\mathbf{u} = \mathbf{v} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{v}\right). \tag{5}$$

By substituting $\mathbf{u} = \mathbf{C}\mathbf{x}$ and $\mathbf{y} = \mathbf{v}$ into (4) and (5), we obtain (2) and (3), respectively.

The GST can be realized using the networks shown in Figs. 1(a) and 1(b). These networks share some similarities with those of the lifting scheme as described in [2]. One difference, however, can be attributed to the fact that we are dealing with N-input N-output networks, where N is potentially larger than two. In such cases, adjacent ladder steps that modify the same channel can be combined, hence, reducing the number of rounding operations and the resulting quantization error [1]. The specific strategy used to realize the transforms $\mathbf{C}$ and $\mathbf{C}^{-1}$ is not particularly critical (from a mathematical standpoint). Since both transforms are linear, each has many equivalent realizations. These two transforms could be implemented using ladder networks, but this is not necessary.

5. CHOICE OF ROUNDING OPERATOR

So far, we have made only one very mild assumption about the rounding operator $\mathcal{Q}$. At this point, we now consider the consequences of a further restriction. Suppose that the rounding operator $\mathcal{Q}$ also satisfies

$$\mathcal{Q}(x+\alpha) = x + \mathcal{Q}(\alpha) \quad \text{for all } x \in \mathbb{Z}, \text{ and all } \alpha \in \mathbb{R}$$

(i.e., $\mathcal{Q}$ is integer-shift invariant). The operator $\mathcal{Q}$ could be chosen, for example, as the floor function (i.e., $\mathcal{Q}(x) = \lfloor x \rfloor$), the ceiling function (i.e., $\mathcal{Q}(x) = \lceil x \rceil$), or a biased floor or ceiling function (i.e., $\mathcal{Q}(x) = \lfloor x + \frac{1}{2} \rfloor$ or $\mathcal{Q}(x) = \lceil x - \frac{1}{2} \rceil$). If $\mathcal{Q}$ is integer-shift invariant, we trivially have the two identities shown in Fig. 2. Therefore, in the case that $\mathcal{Q}$ is integer-shift invariant, we can redraw each of the networks shown in Figs. 1(a) and 1(b) with the rounding unit moved from the input-side to the output-side of the adder. Mathematically, we can rewrite (2) and (3), respectively, as

$$\begin{aligned}\mathbf{y} &= \mathbf{Cx} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{Cx}\right)\\ &= \mathcal{Q}\left(\mathbf{Cx} + (\mathbf{B}-\mathbf{I})\mathbf{Cx}\right)\\ &= \mathcal{Q}\left(\mathbf{BCx}\right),\end{aligned} \tag{6}$$

$$\begin{aligned}\mathbf{x} &= \mathbf{C}^{-1}\left(\mathbf{y} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right)\right)\\ &= -\mathbf{C}^{-1}\left(\mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right) - \mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y} - \mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left((\mathbf{B}-2\mathbf{I})\mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left(-\mathbf{B}^{-1}\mathbf{y}\right).\end{aligned}$$

Or alternately, in the latter case, we can write

$$\mathbf{x} = \mathbf{C}^{-1}\mathcal{Q}'(\mathbf{B}^{-1}\mathbf{y})$$

where

$$\mathcal{Q}'(\alpha) \triangleq -\mathcal{Q}(-\alpha).$$

At this point, we note that $\mathcal{Q}$ and $\mathcal{Q}'$ must be distinct (i.e., different) operators. This follows from the fact that a rounding operator cannot both be integer-shift invariant and satisfy $\mathcal{Q}(\alpha) = -\mathcal{Q}(-\alpha)$ for all $\alpha \in \mathbb{R}$. (See Appendix A for a simple proof.) By observing that $\lfloor -\alpha \rfloor = -\lceil \alpha \rceil$ for all $\alpha \in \mathbb{R}$, we have, for the case of the floor and ceiling operations

$$\mathcal{Q}'(x) = \begin{cases} \lceil x \rceil & \text{for } \mathcal{Q}(x) = \lfloor x \rfloor \\ \lfloor x \rfloor & \text{for } \mathcal{Q}(x) = \lceil x \rceil. \end{cases} \tag{7}$$

The above results are particularly interesting. By examining (6), we can see that all GST-based approximations to a given linear transform (with transform matrix $\mathbf{A} \triangleq \mathbf{BC}$) are exactly equivalent for a fixed choice of $\mathcal{Q}$ (assuming, of course, that $\mathcal{Q}$ is integer-shift invariant). This helps to explain why so many equivalent realizations of the (original) S transform are possible, as this transform utilizes the floor operator which is integer-shift invariant.

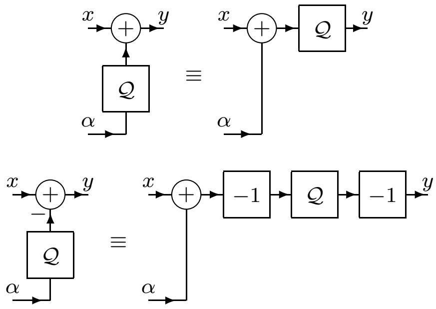

Fig. 2. Transformations for an integer-shift invariant rounding operator $\mathcal{Q}$ ($x \in \mathbb{Z}, \alpha \in \mathbb{R}, y \in \mathbb{Z}$).

6. CALCULATION OF GST PARAMETERS

Suppose that we are given a linear transform characterized by the transform matrix $\mathbf{A}$, and we wish to find a reversible integer-to-integer version of this transform based on the GST framework. In order for this to be possible, we must be able to decompose $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{BC} \tag{8}$$

where the matrices $\mathbf{B}$ and $\mathbf{C}$ are of the forms specified in (2). Therefore, we wish to know which matrices have such a factorization.

For convenience in what follows, let us define

$$\mathbf{A} \triangleq \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N-1,0} & a_{N-1,1} & \cdots & a_{N-1,N-1} \end{bmatrix}.$$

We assert that a factorization of the form of (8) exists if

1. $\mathbf{A}$ is unimodular,
2. $\operatorname{minor}(\mathbf{A}, 0, i)$ is unimodular for some choice of i, $i \in \{0, 1, \ldots, N-1\}$, and
3. $a_{i,j} \in \begin{cases} \mathbb{R} & \text{for } i = 0, j = 0, 1, \ldots, N-1 \\ \mathbb{Z} & \text{for } i = 1, 2, \ldots, N-1, j = 0, 1, \ldots, N-1. \end{cases}$

To prove the above assertion, we begin by considering the slightly more general decomposition

$$\mathbf{A} = \mathbf{BDC} \tag{9}$$

where $\mathbf{B}$ and $\mathbf{C}$ are defined as in (8), and

$$\mathbf{D} \triangleq \begin{bmatrix} b_0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

A comparison of the left- and right-hand sides of (9) yields the trivial relationships

$$c_{i,j} = a_{i,j} \quad \text{for } i = 1, 2, \ldots, N-1, j = 0, 1, \ldots, N-1$$

and the nontrivial system of equations

$$\mathbf{a} = \mathbf{bC}$$

where

$$\mathbf{a} \triangleq \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \end{bmatrix}, \quad \mathbf{b} \triangleq \begin{bmatrix} b_0 & b_1 & \cdots & b_{N-1} \end{bmatrix}.$$

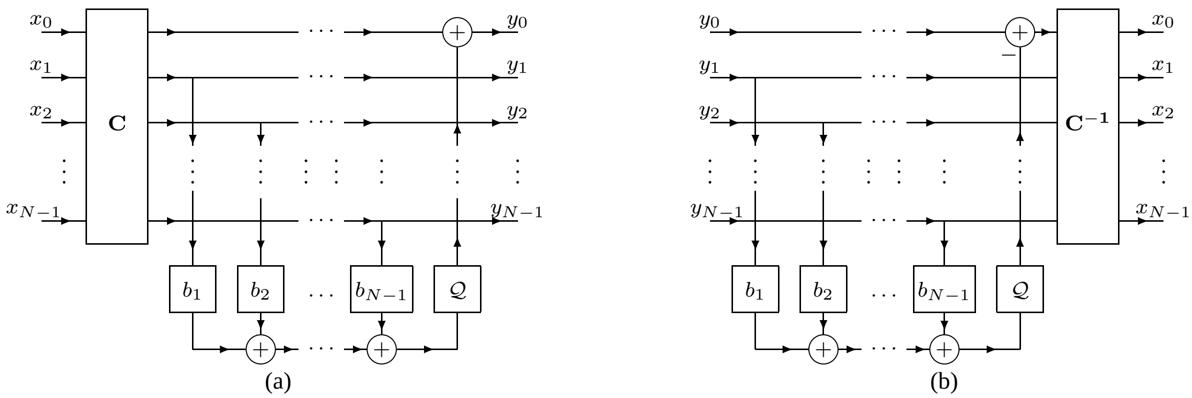

Fig. 1. Network realization of the generalized S transform. (a) Forward transform and (b) inverse transform.

If $\mathbf{C}$ is nonsingular, we can solve for $\mathbf{b}$ in terms of $\mathbf{a}$ as

$$\mathbf{b} = \mathbf{a}\mathbf{C}^{-1}.$$

By considering the determinants of the various matrices in the factorization, we can write

$$\det \mathbf{A} = \det(\mathbf{BDC}) = (\det \mathbf{B})(\det \mathbf{D})(\det \mathbf{C}) = b_0 \det \mathbf{C}.$$

Therefore, $b_0 = (\det \mathbf{A})(\det \mathbf{C})^{-1}$. Consequently, we want to choose $\mathbf{C}$ such that $\det \mathbf{C} = \det \mathbf{A}$. In this case, $b_0 = 1$, so $\mathbf{D} = \mathbf{I}$, and we have $\mathbf{A} = \mathbf{BDC} = \mathbf{BC}$, a factorization of the desired form. Let $w_i \triangleq \det \operatorname{minor}(\mathbf{A}, 0, i)$ for $i = 0, 1, \ldots, N-1$. Using the Laplacian expansion for the determinant of $\mathbf{C}$ across row 0, we obtain

$$\det \mathbf{C} = \textstyle\sum_{i=0}^{N-1} (-1)^i c_{0,i} w_i. \qquad (10)$$

By assumption, $|w_i| = 1$ for some i, say $i = \kappa$. Suppose we choose

$$c_{0,i} = \begin{cases} (-1)^\kappa w_\kappa^{-1} \det \mathbf{A} & \text{for } i = \kappa \\ 0 & \text{otherwise} \end{cases} \qquad (11)$$

for $i = 0, 1, \ldots, N-1$. Substituting (11) into (10), we obtain

$$\det \mathbf{C} = \sum_{i=0}^{N-1} (-1)^i c_{0,i} w_i = (-1)^{2\kappa} w_\kappa w_\kappa^{-1} \det \mathbf{A} = \det \mathbf{A},$$

which is the desired result. Therefore, we can use (11) to generate a valid choice for $\mathbf{C}$, and knowing $\mathbf{C}$, we can solve for $\mathbf{b}$, and hence find $\mathbf{B}$. Thus, we have a constructive proof that the desired factorization of $\mathbf{A}$ exists. Moreover, the factorization is not necessarily unique, since more than one choice may exist for the $\{c_{0,i}\}_{i=0}^{N-1}$ above. In instances where the solution is not unique, we can exploit this degree of freedom in order to minimize computational complexity of the resulting transform realization.

In passing, we note that a more detailed examination of (10) shows that the condition, $|w_i| = 1$ for some i, is overly restrictive. It is only actually necessary that the $\{w_i\}_{i=0}^{N-1}$ not all share a (nontrivial) common factor (i.e., are relatively prime). Clearly, this condition is satisfied if one of the w_i is either plus or minus one. In practice, however, this more general result may only be of limited utility, since the more general case ultimately leads to realizations with greater computational complexity (in most cases).

In all likelihood, the most practically useful transforms in the GST family are those for which the output y_0 is chosen to be a rounded weighted average of the inputs $\{x_i\}_{i=0}^{N-1}$ (with the weights summing to one) and the remaining outputs $\{y_i\}_{i=1}^{N-1}$ are chosen to be any linearly independent set of differences formed from the inputs $\{x_i\}_{i=0}^{N-1}$. Such transforms are particularly useful when the inputs $\{x_i\}_{i=0}^{N-1}$ tend to be highly correlated.

Obviously, there are many ways in which the above set of differences can be chosen. Here, we note that two specific choices facilitate a particularly computationally efficient and highly regular structure for the implementation of $\mathbf{C}$ (and $\mathbf{C}^{-1}$). Suppose the difference outputs are selected in one of two ways:

$$\text{type 1:} \quad y_i = x_i - x_0$$
$$\text{type 2:} \quad y_i = x_i - x_{i-1}$$

for $i = 1, \ldots, N-1$. Since these differences completely define the last $N-1$ rows of $\mathbf{A}$, we can calculate $\det \operatorname{minor}(\mathbf{A}, 0, 0)$. Furthermore, one can easily verify that in both cases $\det \operatorname{minor}(\mathbf{A}, 0, 0) = 1$. Thus, we may choose $\mathbf{C}$ for type-1 and type-2 systems, respectively, as

$$\begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ -1 & 1 & 0 & \ldots & 0 \\ -1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ -1 & 0 & 0 & \ldots & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ -1 & 1 & 0 & \ldots & 0 & 0 \\ 0 & -1 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & 0 & \ldots & -1 & 1 \end{bmatrix}.$$

Furthermore, we can show that the corresponding solutions for $\mathbf{B}$ are, respectively, given by

$$\text{type 1:} \quad b_i = a_{0,i}$$
$$\text{type 2:} \quad b_i = \textstyle\sum_{k=i}^{N-1} a_{0,k}$$

for $i = 1, 2, \ldots, N-1$.

In the above cases, the transform matrices for $\mathbf{C}$ can be realized using the ladder networks shown in Fig. 3. The forward networks for type-1 and type-2 systems are shown in Figs. 3(a) and (b), respectively. The corresponding inverse networks are not shown, due to space constraints, but they are formed simply by reversing the order of the ladder steps, and removing the sign inversion from the adder inputs. For each of the two system types, the forward and inverse networks have the same computational complexity (in terms of the number of arithmetic operations required).

Other choices of differences also facilitate computationally efficient implementations. Choices that yield a $\mathbf{C}$ matrix with all ones on the diagonal are often good in this regard.

7. EXAMPLES

One member of the GST family is, of course, the S transform. We can factor the transform matrix $\mathbf{A}$ as $\mathbf{A} = \mathbf{BC}$ where

$$\mathbf{A} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 1 & -1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

The S transform is obtained by using the parameters from the above factorization for $\mathbf{A}$ in the GST network in conjunction with

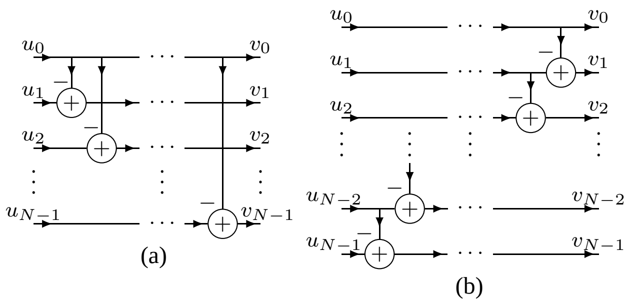

Fig. 3. Networks for realizing particular forms of the $\mathbf{C}$ matrix. The networks for (a) type-1 and (b) type-2 systems.

the rounding operator $\mathcal{Q}(x) = \lfloor x \rfloor$. Mathematically, this gives us

$$\begin{bmatrix} y_0 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_1 + \lfloor \frac{t}{2} \rfloor \\ t \end{bmatrix} \text{ where } t = x_0 - x_1, \text{ and} \tag{12a}$$

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} y_1 + s \\ s \end{bmatrix} \text{ where } s = y_0 - \lfloor \frac{y_1}{2} \rfloor. \tag{12b}$$

Comparing (12b) to (1), we observe that the computational complexity of the former expression is lower (i.e., one less addition is required). Due to our previous results, however, we know that both equations are mathematically equivalent. Thus, we have found a lower complexity implementation of the inverse S transform.

Another example of a transform from the GST family is the reversible color transform (RCT), defined in the JPEG-2000 draft standard [5] (and differing only in minor details from a transform described in [6]). Again, we can factor the transform matrix $\mathbf{A}$ as $\mathbf{A} = \mathbf{BC}$ where

$$\mathbf{A} = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & -1 & 1 \\ 1 & -1 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & \frac{1}{4} & \frac{1}{4} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

The RCT is obtained by using the parameters from the above factorization for $\mathbf{A}$ in the GST network in conjunction with the rounding operator $\mathcal{Q}(x) = \lfloor x \rfloor$. Mathematically, this gives us

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 + \lfloor \frac{1}{4}(t_0 + t_1) \rfloor \\ t_0 \\ t_1 \end{bmatrix} \text{ where } \begin{matrix} t_0 = x_2 - x_1 \\ t_1 = x_0 - x_1 \end{matrix}, \text{ and} \tag{13a}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} s + y_2 \\ s \\ s + y_1 \end{bmatrix} \text{ where } s = y_0 - \lfloor \tfrac{1}{4}(y_1 + y_2) \rfloor. \tag{13b}$$

The formula given for the forward RCT in [5] (see equations G.3–G.5) is

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \lfloor \frac{1}{4}(x_0 + 2x_1 + x_2) \rfloor \\ x_2 - x_1 \\ x_0 - x_1 \end{bmatrix}. \tag{14}$$

By comparing (13a) and (14), we observe that the computational complexity of the former expression is lower (i.e., 4 adds and 1 shift are required instead of, say, 4 adds and 2 shifts). Thus, we have found a lower complexity implementation of the forward RCT. Although the computational complexity is reduced by only one operation, this savings is very significant in relative terms (since only six operations were required before the reduction).

Recall that for integer-shift invariant rounding operators, multiple realization strategies often exist for a particular GST-based reversible integer-to-integer transform. In order to demonstrate this, we now derive an alternative implementation of the RCT. To do this, we factor the transform matrix associated with the RCT (i.e., the matrix $\mathbf{A}$ from above) as $\mathbf{A} = \mathbf{BC}$ where

$$\mathbf{B} = \begin{bmatrix} 1 & \frac{-3}{4} & \frac{1}{4} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 1 \\ 1 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

The corresponding RCT implementation is given by

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_2 + \lfloor \frac{1}{4}(-3t_0 + t_1) \rfloor \\ t_0 \\ t_1 \end{bmatrix} \text{ where } \begin{matrix} t_0 = x_2 - x_1 \\ t_1 = x_0 - x_1 \end{matrix}, \text{ and}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} s_0 + y_2 \\ s_0 \\ s_1 \end{bmatrix} \text{ where } \begin{matrix} s_0 = s_1 - y_1 \\ s_1 = y_0 - \lfloor \frac{1}{4}(-3y_1 + y_2) \rfloor \end{matrix}.$$

One can see that the computational complexity of this alternative implementation is higher than the one proposed in (13). In fact, due to the simple nature of the RCT, the implementation given by (13) is probably the most efficient.

8. CONCLUSIONS

The generalized S transform (GST), a family of reversible integer-to-integer transforms, was proposed. Then, the GST was studied in some detail, leading to a number of interesting results. First, we proved that all GST-based approximations to a given linear transform employing the same integer-shift invariant rounding operator $\mathcal{Q}$ are equivalent. We also showed that the S transform and RCT are specific instances of the GST. Lower complexity implementations of the S transform and RCT were also suggested. Due to the utility of the GST, this family of transforms will no doubt continue to prove useful in both present and future signal coding applications.

A. PROOFS

Lemma A.1. *A rounding operator $\mathcal{Q}$ that is integer-shift invariant cannot also possess the antisymmetry property (i.e., $\mathcal{Q}(\alpha) = -\mathcal{Q}(-\alpha)$ for all $\alpha \in \mathbb{R}$).*

Proof. Consider the quantity $\mathcal{Q}(\frac{1}{2})$. Using trivial algebraic manipulation and the integer-shift invariance property, we have

$$\mathcal{Q}(\tfrac{1}{2}) = \mathcal{Q}(1 - \tfrac{1}{2}) = 1 + \mathcal{Q}(-\tfrac{1}{2}). \tag{16}$$

From the antisymmetry property, we can write

$$\mathcal{Q}(\tfrac{1}{2}) = -\mathcal{Q}(-\tfrac{1}{2}). \tag{17}$$

Combining (16) and (17), we obtain

$$1 + \mathcal{Q}(-\tfrac{1}{2}) = -\mathcal{Q}(-\tfrac{1}{2}) \Rightarrow \mathcal{Q}(-\tfrac{1}{2}) = \tfrac{1}{2}.$$

Thus, we have that $\mathcal{Q}(\frac{1}{2}) \notin \mathbb{Z}$. Since, by definition, $\mathcal{Q}$ must always yield an integer result, the integer-shift invariance and antisymmetry properties cannot be simultaneously satisfied by $\mathcal{Q}$. □

B. REFERENCES

[1] M. D. Adams, "Reversible wavelet transforms and their application to embedded image compression," M.A.Sc. thesis, Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada, Jan. 1998, Available from `http://www.ece.ubc.ca/~mdadams`.

[2] A. R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet transforms that map integers to integers," *Applied and Computational Harmonic Analysis*, vol. 5, no. 3, pp. 332–369, July 1998.

[3] A. Said and W. A. Pearlman, "An image multiresolution representation for lossless and lossy compression," *IEEE Trans. on Image Processing*, vol. 5, no. 9, pp. 1303–1310, Sept. 1996.

[4] A. Zandi, J. D. Allen, E. L. Schwartz, and M. Boliek, "CREW: Compression with reversible embedded wavelets," in *Proc. of IEEE Data Compression Conference*, Snowbird, UT, USA, Mar. 1995, pp. 212–221.

[5] ISO/IEC, *ISO/IEC FDIS 15444-1, Information technology - JPEG 2000 image coding system*, Sept. 2000.

[6] M. J. Gormish, E. L. Schwartz, A. F. Keith, M. P. Boliek, and A. Zandi, "Lossless and nearly lossless compression of high-quality images," in *Proc. of SPIE*, San Jose, CA, USA, Mar. 1997, vol. 3025, pp. 62–70.