Generalized S Transform

Michael D. Adams, *Member, IEEE,* Faouzi Kossentini, *Senior Member, IEEE,* and Rabab Ward, *Fellow, IEEE*

Dept. of Elec. and Comp. Engineering, University of British Columbia,
Vancouver, BC, Canada V6T 1Z4

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 detail, by considering topics such as GST parameter calculation, the effects of using different rounding operators in the GST, and the relationship between the GST and the lifting scheme. Some examples of specific transforms in the GST family are also given. In particular, a new transform in this family is introduced, and its practical utility demonstrated.

EDICS Category: 3-COMP (Signal Representation and Compression)

Please address all correspondence concerning this manuscript to:
Michael D. Adams
Department of Electrical and Computer Engineering
University of British Columbia
2356 Main Mall
Vancouver, BC, Canada V6T 1Z4
Phone: 604-822-2872, Fax: 604-822-5949
Email: `mdadams@ieee.org`

Generalized S Transform[*]

Michael D. Adams, *Member, IEEE*, Faouzi Kossentini, *Senior Member, IEEE*, and Rabab Ward, *Fellow, IEEE*

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

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 detail, by considering topics such as GST parameter calculation, the effects of using different rounding operators in the GST, and the relationship between the GST and the lifting scheme. Some examples of specific transforms in the GST family are also given. In particular, a new transform in this family is introduced, and its practical utility demonstrated.

I. Introduction

Reversible integer-to-integer transforms have become a popular tool for use in signal coding applications requiring lossless signal reproduction [1–5]. One of the best known transforms of this type is the S transform [3,4,6]. In this manuscript, 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) and reversible integer-to-integer transforms in general.

The remainder of this manuscript is structured as follows. We begin, in Section II, with a brief discussion of the notation and terminology used herein. The S transform is then introduced in Section III, and the GST family of transforms is defined in Section IV. Sections V and VI proceed to examine GST parameter calculation and the effects of using different rounding operators in the GST. Some examples of well known transforms belonging to the GST family are given in Section VII, and in Section VIII, we present a new GST-based transform, and demonstrate its utility for image coding applications. Finally, we conclude in Section IX with a summary of our results and some closing remarks.

II. Notation and Terminology

Before proceeding further, a short digression concerning the notation and terminology used in this manuscript 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 $\mathbf{A}$. The notation $\Pr(x)$ denotes the probability of event x, and the notation $\operatorname{E}(x)$ denotes the expected value of the quantity x.

Suppose that $\mathcal{Q}$ denotes a rounding operator. In this manuscript, 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. A rounding operator $\mathcal{Q}$ is said to be integer invariant if it satisfies

$$\mathcal{Q}(x) = x \quad \text{for all } x \in \mathbb{Z} \tag{1}$$

(i.e., $\mathcal{Q}$ leaves integers unchanged). As customary, in this manuscript, we consider only integer-invariant rounding operators. For obvious reasons, rounding operators that are not integer invariant are of little practical value. At times, we focus our attention on six specific rounding operators (i.e., the floor, biased floor, ceiling, biased ceiling, truncation, and biased truncation functions) which are defined below.

For $r \in \mathbb{R}$, the notation $\lfloor r \rfloor$ denotes the largest integer not more than r (i.e., the floor function), and the notation $\lceil r \rceil$ denotes the smallest integer not less than r (i.e., the ceiling function). In passing, we also note that the floor and ceiling functions satisfy the relationship

$$\lceil r \rceil = -\lfloor -r \rfloor \quad \text{for all } r \in \mathbb{R}. \tag{2}$$

The biased floor, biased ceiling, truncation, and biased truncation functions are defined, respectively, as

$$\operatorname{bfloor} r = \left\lfloor r + \tfrac{1}{2} \right\rfloor, \quad \operatorname{bceil} r = \left\lceil r - \tfrac{1}{2} \right\rceil, \tag{3}$$

$$\operatorname{trunc} r = \begin{cases} \lfloor r \rfloor & \text{for } r \geq 0 \\ \lceil r \rceil & \text{for } r < 0, \end{cases} \quad \text{and} \tag{4}$$

$$\operatorname{btrunc} r = \begin{cases} \operatorname{bfloor} r & \text{for } r \geq 0 \\ \operatorname{bceil} r & \text{for } r < 0. \end{cases} \tag{5}$$

The mapping associated with the btrunc function is essentially equivalent to traditional rounding (to the nearest integer). The fractional part of a real number r is denoted $\operatorname{frac} r$ and defined as

$$\operatorname{frac} r \stackrel{\triangle}{=} r - \lfloor r \rfloor .$$

Thus, we have that $0 \leq \operatorname{frac} r < 1$ for all $r \in \mathbb{R}$. We define the mod function as

$$\operatorname{mod}(x, y) \stackrel{\triangle}{=} x - y \lfloor x/y \rfloor \quad \text{where } x \in \mathbb{Z}, y \in \mathbb{Z}. \tag{6}$$

This function simply computes the remainder obtained when x is divided by y, with division being defined in such a way that the remainder is always nonnegative. From (6) and (2), we trivially have the identities

$$\lfloor x/y \rfloor = \tfrac{x - \operatorname{mod}(x,y)}{y} \quad \text{for } y \neq 0 \tag{7}$$

$$\lceil x/y \rceil = \tfrac{x + \operatorname{mod}(-x,y)}{y} \quad \text{for } y \neq 0. \tag{8}$$

III. S Transform

One of the simplest and most ubiquitous reversible integer-to-integer mappings is the S transform [2–4], 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 [2–4]. 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., equation (1) in [3] and equation (3.3) in [4]) as

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

The corresponding inverse transform is given (e.g., equation (2) in [3]) by

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} t \\ t-y_1 \end{bmatrix}, \text{ where } t \triangleq y_0 + \left\lfloor \tfrac{1}{2}(y_1+1) \right\rfloor. \tag{9}$$

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.

IV. 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

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

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

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{C}\mathbf{x}\right), \tag{10}$$

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

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

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

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

and $\mathcal{Q}$ is a rounding operator. By examining (10), 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 $\mathbf{A} \overset{\triangle}{=} \mathbf{BC}$. Thus, the GST can be viewed as a reversible integer-to-integer mapping that approximates the linear transform characterized by the matrix $\mathbf{A}$. The inverse GST is given by

$$\mathbf{x} = \mathbf{C}^{-1}\left(\mathbf{y} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right)\right). \tag{11}$$

Note that $\mathbf{C}^{-1}$ is an integer matrix, since, by assumption, $\mathbf{C}$ is a unimodular integer matrix. To show that (11) is, in fact, the inverse of (10), one need only observe that due to the form of $\mathbf{B}$, for any two $N \times 1$ vectors $\mathbf{u}$ and $\mathbf{v}$:

$$\mathbf{v} = \mathbf{u} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{u}\right) \tag{12}$$

implies

$$\mathbf{u} = \mathbf{v} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{v}\right). \tag{13}$$

By substituting $\mathbf{u} = \mathbf{Cx}$ and $\mathbf{v} = \mathbf{y}$ into (12) and (13), we obtain (10) and (11), respectively.

The GST can be realized using the networks shown in Figs. 1(a) and 1(b). These networks share some similarities with the reversible integer-to-integer ladder networks described in [2, 5] (which, in turn, are related to ideas in [7, 8]). This said, however, some differences also exist, as explained below.

The first difference 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, reducing the number of rounding operations and resulting quantization error. For example, suppose that we have a N-input N-output network comprised of $(N-1)$ ladder steps, all of which modify the 0th channel. Although we could naively perform rounding separately for each ladder step, as shown in Fig. 2(a), the rounding error can be significantly reduced by grouping the ladder steps together and performing rounding only once, as shown in Fig. 2(b).

The second difference relates to the presence of the unimodular integer matrix $\mathbf{C}$ (or $\mathbf{C}^{-1}$). In the case of the lifting scheme, filtering is performed exclusively by a ladder network, while with the GST framework, filtering is accomplished by the cascade of a ladder network and a linear network with a unimodular integer transfer matrix (i.e., $\mathbf{C}$ or $\mathbf{C}^{-1}$). This difference, however, is arguably less important, since a unimodular integer matrix can always be decomposed into a product of elementary unimodular integer matrices and such a factorization is essentially a lifting factorization. (The proof that a unimodular integer matrix can be decomposed into a product of elementary unimodular integer matrices is constructive by the standard algorithm for computing the Smith normal form of an integer matrix (e.g., as described in [9]).)

From a mathematical viewpoint, the specific strategy used to realize the transforms $\mathbf{C}$ and $\mathbf{C}^{-1}$ is not particularly critical. Since both transforms are linear, each has many equivalent realizations. These two transforms could be implemented using ladder networks, as suggested above, but this is not necessary.

In passing, we would like to note that Hao and Shi [10] have done some related work with N-point block transforms. Their work is concerned primarily with the lifting factorization problem for general block transforms, whereas we study in detail a very specific class of these transforms (i.e., those belonging to the GST family). Thus, the results presented herein and those presented in [10] are complementary. Other works, such as [1, 7], have also considered reversible N-input N-output ladder networks in various contexts (where $N \geq 2$). These other works, however, do not specifically consider the GST family of transforms. Thus, our results are distinct from those in the preceding works.

V. 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 approximation to this transform based on the GST framework. In order for this to be possible, we must be able to decompose $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{B}\mathbf{C} \tag{14}$$

where the matrices $\mathbf{B}$ and $\mathbf{C}$ are of the forms specified in (10). Therefore, we wish to know which matrices have such a factorization. The answer to this question is given by the theorem below.

Theorem 1 (Existence of GST Factorization) The real matrix $\mathbf{A}$, defined as

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

has a GST factorization (i.e., a factorization of the form of (14)) if and only if all of the following conditions hold:

1. The last $N-1$ rows of $\mathbf{A}$ must contain only integer entries. That is,

$$a_{i,j} \in \mathbb{Z} \quad \text{for } i = 1, 2, \ldots, N-1, j = 0, 1, \ldots, N-1.$$

2. $\mathbf{A}$ is unimodular.
3. The integers $\{\det \operatorname{minor}(\mathbf{A}, 0, i)\}_{i=0}^{N-1}$ are relatively prime.

Proof: First, we prove the sufficiency of the above conditions. We begin by considering the slightly more general decomposition

$$\mathbf{A} = \mathbf{B}\mathbf{D}\mathbf{C} \tag{15}$$

where $\mathbf{B}$ and $\mathbf{C}$ are defined as in (14), and

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

A comparison of the left- and right-hand sides of (15) yields the trivial relationships

$$c_{i,j} = a_{i,j} \quad \text{for } i = 1, 2, \dots, N-1, j = 0, 1, \dots, N-1 \tag{16}$$

and the nontrivial system of equations

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

where

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

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

$$\mathbf{b} = \mathbf{aC}^{-1}.$$

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, $b_0 = (\det \mathbf{A})/(\det \mathbf{C})$. 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 $i = 0, 1, \dots, N-1$. From (16), we know

$$w_i = \det \operatorname{minor}(\mathbf{C}, 0, i) \tag{17}$$

for $i = 0, 1, \dots, N-1$. Using the Laplacian expansion for the determinant of $\mathbf{C}$ across row 0, we obtain

$$\det \mathbf{C} = \textstyle\sum_{i=0}^{N-1} (-1)^i c_{0,i} w_i. \tag{18}$$

Since $\det \mathbf{A} \in \{-1, 1\}$ and we want to choose the $\{c_{0,i}\}_{i=0}^{N-1}$ such that $\det \mathbf{C} = \det \mathbf{A}$, from (18) we must solve

$$1 = \textstyle\sum_{i=0}^{N-1} c'_{0,i} w'_i \tag{19}$$

where $c'_{0,i} = c_{0,i}/\det \mathbf{A}$ and $w'_i = (-1)^i w_i$ for $i = 0, 1, \dots, N-1$. In this equation, the $\{w'_i\}_{i=0}^{N-1}$ are relatively prime since the $\{w_i\}_{i=0}^{N-1}$ are (by assumption) relatively prime, and the $\{c'_{0,i}\}_{i=0}^{N-1}$ are integers since the $\{c_{0,i}\}_{i=0}^{N-1}$

are integers and $\det \mathbf{A} \in \{-1, 1\}$. We can show that (19) always has a solution. This result follows immediately from the well known result in number theory that states: The greatest common divisor (GCD) of a set of integers is a linear combination (with integer coefficients) of those integers, and moreover, it is the smallest positive linear combination of those integers [11]. In our case, the integers $\{w'_i\}_{i=0}^{N-1}$ are relatively prime, so their GCD is 1, and therefore, 1 must be a linear combination of the integers $\{w'_i\}_{i=0}^{N-1}$. Furthermore, the coefficients of the linear combination can be found using the Euclidean algorithm [11]. Therefore, we can use (19) 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. This shows the sufficiency of the above conditions on the matrix $\mathbf{A}$.

The necessity of the above conditions on the matrix $\mathbf{A}$ is easily shown. Due to the form of $\mathbf{B}$, the last $N-1$ rows of the matrices $\mathbf{A}$ and $\mathbf{C}$ must be the same. Thus, the last $N-1$ rows of $\mathbf{A}$ must contain only integer entries, since $\mathbf{C}$ is an integer matrix. If $\mathbf{A}$ is not unimodular, obviously it cannot be decomposed into a product of two unimodular matrices (namely, $\mathbf{B}$ and $\mathbf{C}$). This follows from the fact that $\det \mathbf{A} = \det \mathbf{B} \det \mathbf{C}$, $\det \mathbf{B} \in \{-1, 1\}$, and $\det \mathbf{C} \in \{-1, 1\}$. The relative primeness of $\{\det \operatorname{minor}(\mathbf{A}, 0, i)\}_{i=0}^{N-1}$ follows from the "GCD is a linear combination" theorem stated above. If this set of integers is not relatively prime, their GCD is greater than one, and no solution to (19) can exist. Thus, the stated conditions on the matrix $\mathbf{A}$ are necessary for a GST factorization to exist. ■

From the proof of the above theorem, we know that 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 the computational complexity of the resulting transform realization.

For most practically useful transforms in the GST family, it is often the case that $\{\operatorname{minor}(\mathbf{A}, 0, i)\}_{i=0}^{N-1}$ are all unimodular (i.e., $|\det \operatorname{minor}(\mathbf{A}, 0, i)| = 1$ for all $i \in \{0, 1, \ldots, N-1\}$). This condition holds for all of the examples considered later in Section VII. In such instances, the most useful GST factorizations are often obtained by choosing the unknowns $\{c_{0,i}\}_{i=0}^{N-1}$ so that all but one are zero, with the remaining unknown being chosen as either -1 or 1 in order to satisfy (19). In this way, we can readily obtain N distinct GST factorizations of $\mathbf{A}$ which typically yield low complexity transform realizations.

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

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

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

In the above cases, the transform matrices for $\mathbf{C}$ and $\mathbf{C}^{-1}$ can be realized using the ladder networks shown in Fig. 3. The forward and inverse networks for a type-1 system are shown in Figs. 3(a) and (b), respectively. Similarly, the forward and inverse networks for a type-2 system are shown in Figs. 3(c) and (d). 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.

VI. Choice of Rounding Operator

So far, we have made only one very mild assumption about the rounding operator $\mathcal{Q}$ (i.e., property (1) holds). At this point, we now consider the consequences of a further restriction. Suppose that the rounding operator $\mathcal{Q}$ also satisfies

$$\mathcal{Q}(x + \alpha) = x + \mathcal{Q}(\alpha) \quad \text{for all } x \in \mathbb{Z}, \text{ and all } \alpha \in \mathbb{R} \tag{20}$$

(i.e., $\mathcal{Q}$ is integer-shift invariant). The operator $\mathcal{Q}$ could be chosen, for example, as the floor, biased floor, ceiling, or biased ceiling operator (as defined in Section II). Note that the truncation and biased truncation operators are not integer-shift invariant. To demonstrate this for either operator, one can simply evaluate the expressions on the left- and right-hand sides of equation (20) for $\alpha = -\frac{1}{2}$ and $x = 1$. Clearly, the two expressions are not equal (i.e., $0 \neq 1$ or $1 \neq 0$).

If $\mathcal{Q}$ is integer-shift invariant, we trivially have the two identities shown in Fig. 4. 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 (10) and (11), respectively, as

$$\begin{aligned}\mathbf{y} &= \mathbf{Cx} + \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{Cx}\right) \qquad (21a)\\ &= \mathcal{Q}\left(\mathbf{Cx} + (\mathbf{B}-\mathbf{I})\mathbf{Cx}\right)\\ &= \mathcal{Q}\left(\mathbf{BCx}\right), \text{ and}\end{aligned}$$

$$\begin{aligned}\mathbf{x} &= \mathbf{C}^{-1}\left(\mathbf{y} - \mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right)\right) \qquad (21b)\\ &= -\mathbf{C}^{-1}\left(\mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y}\right) - \mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left((\mathbf{B}-\mathbf{I})\mathbf{y} - \mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left((\mathbf{B}-2\mathbf{I})\mathbf{y}\right)\\ &= -\mathbf{C}^{-1}\mathcal{Q}\left(-\mathbf{B}^{-1}\mathbf{y}\right).\end{aligned}$$

Or alternately, in the latter case, we can write

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

where

$$\mathcal{Q}'(\alpha) \stackrel{\triangle}{=} -\mathcal{Q}(-\alpha).$$

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 I for a simple proof.) By using (2), we have, for the case of the floor and ceiling operations

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

The above results are particularly interesting. Recall that the GST approximates a linear transform characterized by the matrix $\mathbf{A} \stackrel{\triangle}{=} \mathbf{BC}$. Examining the definition of the forward GST given by (10), we can see that, due to the rounding operator $\mathcal{Q}$, the underlying transform is a function of both $\mathbf{B}$ and $\mathbf{C}$ individually, and not just their product $\mathbf{A}$. In other words, different factorizations of $\mathbf{A}$ (into $\mathbf{B}$ and $\mathbf{C}$) do not necessarily yield equivalent transforms. When $\mathcal{Q}$ is integer-shift invariant, however, (10) simplifies to (21a), in which case the underlying transform depends only on the product $\mathbf{A}$. Thus, for a fixed choice of integer-shift invariant operator $\mathcal{Q}$, all GST-based approximations to a given linear transform (characterized by $\mathbf{A}$) are exactly equivalent. 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.

Computational Complexity

As one might suspect, the choice of rounding operator can affect computational complexity. In what follows, we examine the effects of this choice for six different rounding operators (i.e., the floor, biased floor, ceiling, biased ceiling, truncation, and biased truncation operators).

Consider the network of Fig. 2(b), the fundamental building block of the GST and also a much larger class of reversible integer-to-integer N-input N-output mappings. This network simply computes the quantity

$$v_0 = u_0 + \mathcal{Q}(\textstyle\sum_{i=1}^{N-1} b_i u_i). \tag{22}$$

Without loss of generality[1], we assume that the gains $\{b_i\}_{i=1}^{N-1}$ can be expressed exactly as dyadic rational numbers (i.e., numbers of the form $\frac{x}{2^F}$, where $x \in \mathbb{Z}$, $F \in \mathbb{Z}$, and $F \geq 1$). Provided that F is chosen large enough, we can always define $\hat{b}_i \overset{\triangle}{=} 2^F b_i$ for $i = 1, 2, \ldots, N-1$ where the $\{\hat{b}_i\}_{i=1}^{N-1} \in \mathbb{Z}$. Often, in practice, we need to compute the expression given by (22) using integer arithmetic. In such cases, we typically calculate

$$v_0 = u_0 + \mathcal{Q}\left(\tfrac{1}{2^F} \textstyle\sum_{i=1}^{N-1} \hat{b}_i u_i\right).$$

Thus, we must compute an expression of the form

$$\mathcal{Q}\left(\tfrac{x}{2^F}\right) \tag{23}$$

where $x \in \mathbb{Z}$.

Let us now consider the calculation of an expression of the form of (23) for each of the six rounding operators

[1] Any given real number can be approximated with arbitrary accuracy by an appropriately chosen dyadic rational number.

mentioned above. Assuming a two's complement integer representation, we can show that

$$\left\lfloor \tfrac{x}{2^F} \right\rfloor = \operatorname{asr}(x, F), \quad \operatorname{bfloor} \tfrac{x}{2^F} = \operatorname{asr}(x + H, F),$$

$$\left\lceil \tfrac{x}{2^F} \right\rceil = \operatorname{asr}(x + M, F), \quad \operatorname{bceil} \tfrac{x}{2^F} = \operatorname{asr}(x + B, F),$$

$$\operatorname{trunc}(\tfrac{x}{2^F}) = \begin{cases} \operatorname{asr}(x, F) & \text{for } x \geq 0 \\ \operatorname{asr}(x + M, F) & \text{for } x < 0 \end{cases} \quad \text{and}$$

$$\operatorname{btrunc}(\tfrac{x}{2^F}) = \begin{cases} \operatorname{asr}(x + H, F) & \text{for } x \geq 0 \\ \operatorname{asr}(x + B, F) & \text{for } x < 0. \end{cases}$$

where $M = 2^F - 1$, $H = 2^{F-1}$, $B = 2^{F-1} - 1$, and $\operatorname{asr}(x, n)$ denotes the arithmetic right shift of x by n bits. The above expressions can be derived by observing that $\operatorname{asr}(x, n) = \left\lfloor \frac{x}{2^n} \right\rfloor$ and $\lceil x/y \rceil = \lfloor (x + y - 1)/y \rfloor$. (See Appendix II for proofs of these assertions.) From above, we can see that the floor operator incurs the least computation (i.e., one shift). The biased floor, ceiling, and biased ceiling operators require slightly more computation (i.e., one addition and one shift in total). Finally, the truncation and biased truncation operators are the most complex due to their behavioral dependence on the sign of the operand.

Rounding Error

When constructing reversible integer-to-integer versions of linear transforms, one must remember that the effects of rounding error can be significant. The objective is to produce a (nonlinear) integer-to-integer mapping that closely approximates some linear transform with desirable properties. In order to preserve the desirable properties of the parent linear transform, it is usually beneficial to minimize the error introduced by rounding intermediate results to integers.

In the case of the GST, only one network of the form shown in Fig. 2(b) is employed. Thus, rounding is only performed once, and the impact is relatively small. If, however, multiple networks of this form are cascaded together (as in the case of more general lifting-based frameworks), rounding error may be more of a concern. Therefore, in some applications, it may be desirable to choose the rounding operator in order to minimize rounding error.

Suppose that we must round real numbers of the form $\frac{x}{2^F}$ where $x \in \mathbb{Z}$, $F \in \mathbb{Z}$, and $F \geq 1$. Such numbers are associated with a fixed-point representation having F fraction bits (i.e., F bits to the right of the binary point). If we assume that, for both negative and nonnegative quantities, all 2^F representable fractional values are equiprobable, one can show that the error interval, peak absolute error (PAE), and mean absolute error (MAE) for each of the rounding operators under consideration are as listed in Table I. (See Appendix III for the derivation of the error intervals and MAE expressions. The PAE expressions follow immediately from the error intervals.) By examining the table, one can easily see that, for $F \geq 2$, the biased rounding operators have the smallest MAEs and PAEs. Consequently, the use of the biased operators has the advantage of reduced rounding error. It is important to note, however, that the preceding assertion does not hold if $F = 1$. (Typically, we would have $F = 1$ when the $\{b_i\}_{i=1}^{N-1}$ in (22) are all integer multiples of one half.) If $F = 1$, all of the rounding operators under consideration have the same PAE (i.e., $\frac{1}{2}$) and the same MAE (i.e., $\frac{1}{4}$), and consequently, all of these operators are equally good in terms of rounding error. In practice, the case of $F = 1$ is not so unlikely to occur. Consequently, it is important to note the peculiar behavior in this case. In fact, some well known papers on reversible integer-to-integer transforms have overlooked this fact (and used a bias of one half when it is not benefi cial to do so).

VII. 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

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

The S transform is obtained by using the parameters from the above factorization for $\mathbf{A}$ in the GST network in conjunction with the rounding operator $\mathcal{Q}(x) = \lfloor x \rfloor$. (The particular network used to realize the matrix $\mathbf{C}$ is indicated by the factorization of $\mathbf{C}$ shown above.) Mathematically, this gives us

$$\begin{bmatrix} y_0 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_1 + \left\lfloor \frac{t}{2} \right\rfloor \\ t \end{bmatrix} \text{ where } t = x_0 - x_1, \text{ and} \tag{24a}$$

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} y_1 + s \\ s \end{bmatrix} \text{ where } s = y_0 - \left\lfloor \frac{y_1}{2} \right\rfloor. \tag{24b}$$

Comparing (9) to (24b), we observe that the computational complexity of the latter 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), defi ned in the JPEG-2000 Part-1 standard [12] (and differing only in minor details from a transform described in [13]). 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 $\mathcal{Q}(x) = \lfloor x \rfloor$. (Again, the particular network used to realize the matrix $\mathbf{C}$ is indicated by the factorization of $\mathbf{C}$ shown above.) Mathematically, this gives us

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 + \left\lfloor \frac{1}{4}(t_0+t_1) \right\rfloor \\ t_0 \\ t_1 \end{bmatrix} \text{ where } \begin{smallmatrix} t_0 = x_2 - x_1 \\ t_1 = x_0 - x_1 \end{smallmatrix}, \text{ and} \tag{25a}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} s+y_2 \\ s \\ s+y_1 \end{bmatrix} \text{ where } s = y_0 - \left\lfloor \tfrac{1}{4}(y_1 + y_2) \right\rfloor. \tag{25b}$$

The formula given for the forward RCT in [12] is

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \left\lfloor \frac{1}{4}(x_0+2x_1+x_2) \right\rfloor \\ x_2 - x_1 \\ x_0 - x_1 \end{bmatrix}. \tag{26}$$

By comparing (25a) and (26), we observe that the computational complexity of the former expression is lower (i.e., four adds and one shift are required instead of, say, four adds and two 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 alternative RCT implementation is obtained by using the parameters from the above factorization for $\mathbf{A}$ in the GST network in conjunction with the rounding operator $\mathcal{Q}(x) = \lfloor x \rfloor$. (Again, the particular network used to realize the matrix $\mathbf{C}$ is indicated by the factorization of $\mathbf{C}$ shown above.) The corresponding RCT implementation is given by

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

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

One can see that the computational complexity of this alternative implementation is higher than the one proposed in (25). In fact, it is likely that, due to the simple nature of the RCT, the implementation given by (25) is probably the most efficient.

VIII. Practical Application

When compressing a color image, a multicomponent transform is often applied to the color components in order to improve coding efficiency. Such a transform is frequently defined so that one of its output components approximates luminance. In this way, one can easily extract a grayscale version of a coded image if so desired. For RGB color imagery, luminance is related to the color component values by

$$y = 0.299r + 0.587g + 0.114b \tag{28}$$

where y is the luminance value, and r, g, and b are the red, green, and blue component values, respectively.

The forward RCT is defined in such a way that one of its output components approximates luminance. In effect, the RCT computes the luminance as $\frac{1}{4}r + \frac{1}{2}g + \frac{1}{4}b$. Obviously, this is a rather crude approximation to the true luminance as given by (28). Without a huge sacrifice in complexity, however, we can define a new RCT-like transform that provides a much better approximation to true luminance. To this end, we propose a slightly altered version of the RCT called the modified RCT (MRCT).

The forward and inverse equations for the MRCT are given, respectively, by

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_0 + \left\lfloor \frac{1}{128}(90t_0 + 15t_1) \right\rfloor \\ t_0 \\ t_1 \end{bmatrix} \text{ where } \begin{matrix} t_0 = x_1 - x_0 \\ t_1 = x_2 - x_1 \end{matrix} \quad \text{and} \tag{29a}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} s_0 \\ s_1 \\ s_1 + y_2 \end{bmatrix} \text{ where } \begin{matrix} s_0 = y_0 - \left\lfloor \frac{1}{128}(90y_1 + 15y_2) \right\rfloor \\ s_1 = s_0 + y_1. \end{matrix} \tag{29b}$$

The MRCT is associated with the transform matrix

$$\mathbf{A} = \begin{bmatrix} \frac{38}{128} & \frac{75}{128} & \frac{15}{128} \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix},$$

which can be factored as $\mathbf{A} = \mathbf{BC}$, where

$$\mathbf{B} = \begin{bmatrix} 1 & \frac{90}{128} & \frac{15}{128} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}.$$

The MRCT is obtained by using the above factorization for $\mathbf{A}$ in the GST network in conjunction with the rounding operator $\mathcal{Q}(x) = \lfloor x \rfloor$. (The particular network used to realize the matrix $\mathbf{C}$ is indicated by the factorization of $\mathbf{C}$ shown above.) The forward and inverse MRCT can each be implemented with six adds and four shifts (where expressions of the form $\left\lfloor \frac{1}{128}(90u + 15v) \right\rfloor$ can be computed as $\operatorname{asr}((16 - 1)[(4 + 2)u + v], 7)$). Recall that the forward and inverse RCT, as given by (25), each require four adds and one shift. Thus, in terms of computational complexity, the MRCT compares quite favorably with the RCT, when one considers the RCT's extreme simplicity.

Having defined the MRCT, we first studied its coding performance. A set of 196 RGB color images, covering a wide variety of content, was compressed losslessly using both the MRCT and RCT in the same coding system.

The difference in the overall bit rate obtained with the two transforms was less than 0.1%. Thus, the MRCT and RCT are quite comparable in terms of coding efficiency.

In addition to coding efficiency, we also examined the luminance approximation quality. For each image, we computed the true luminance component as given by (28) and the luminance components generated by the MRCT and RCT. We then proceeded to calculate the approximation error for each transform. The results for several of the test images are shown in Table II. Clearly, the MRCT approximates true luminance much more accurately than the RCT (in terms of both MAE and PAE). Furthermore, this difference in approximation quality is usually visually perceptible. That is, the MRCT yields a luminance component that appears visually identical to the true luminance component, while the RCT generates a luminance component that is often visibly different from the ideal.

When designing the MRCT, we had to choose which set of differences to employ (for the two non-luminance outputs of the forward transform). In passing, we would like to note that this choice can impact coding efficiency. For natural imagery, experimental results suggest that the difference between the red and blue components is often slightly more difficult to code than the other two differences (i.e., green-blue and green-red). For this reason, we chose to use the green-blue and green-red differences to form the non-luminance outputs of the (forward) MRCT.

IX. 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. We proved that for a fixed choice of integer-shift invariant rounding operator, all GST-based approximations to a given linear transform are equivalent. Also, we studied several common rounding operators, examining their computational complexity and rounding error properties. When considering both computational complexity and rounding error characteristics together, the floor, biased floor, ceiling, and biased ceiling operators are arguably the most practically useful of those considered. The floor operator is desirable as it has the lowest computational complexity, while the biased floor and biased ceiling operators are slightly more computationally complex but normally have the best rounding error performance.

The S transform and RCT were shown to be specific instances of the GST. Lower complexity implementations of the S transform and RCT were also suggested. A new transform belonging to the GST family, called the MRCT, was introduced, and shown to be practically useful for image coding applications. Due to its utility, the GST family of transforms will no doubt continue to prove useful in both present and future signal coding applications.

APPENDICES

I. ANTISYMMETRY AND INTEGER-SHIFT INVARIANCE

Proposition 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

$$\mathcal{Q}(\tfrac{1}{2}) = \mathcal{Q}(1 - \tfrac{1}{2}) = 1 + \mathcal{Q}(-\tfrac{1}{2}) \tag{30}$$

From the antisymmetry property, we can write

$$\mathcal{Q}(\tfrac{1}{2}) = -\mathcal{Q}(-\tfrac{1}{2}) \tag{31}$$

Combining (30) and (31), 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 $\mathcal{Q}$. ■

II. RELATIONSHIPS INVOLVING THE FLOOR OR CEILING FUNCTION

Proposition 2: Suppose that x is an integer represented in two's complement form with a word size of P bits. Let $\operatorname{asr}(x, n)$ denote the quantity resulting from the arithmetic right shift of x by n bits. We assert that

$$\operatorname{asr}(x, n) = \left\lfloor \tfrac{x}{2^n} \right\rfloor \quad \text{where } 0 \leq n \leq P - 1 \tag{32}$$

Proof: Let $a_0, a_1, \ldots, a_{P-1}$ denote the P bits of the word representing x, where a_0 and a_{P-1} are the least- and most-significant bits, respectively. In two's complement form, the integer x is represented as

$$x = -a_{P-1}2^{P-1} + \textstyle\sum_{i=0}^{P-2} a_i 2^i \tag{33}$$

Through the properties of modular arithmetic, one can show that

$$\operatorname{mod}(x, 2^n) = \textstyle\sum_{i=0}^{n-1} a_i 2^i \quad \text{for } 0 \leq n \leq P - 1. \tag{34}$$

From the definition of the arithmetic shift right operation, we can write (for $0 \leq n \leq P-1$)

$$\begin{aligned}\operatorname{asr}(x,n) &= -a_{P-1}2^{P-1} + a_{P-1}\left(\sum_{i=P-1-n}^{P-2} 2^i\right) + \\ &\quad \sum_{i=0}^{P-2-n} a_{i+n}2^i \\ &= -a_{P-1}(2^{P-1} - 2^{P-1} + 2^{P-1-n}) + \\ &\quad \sum_{i=0}^{P-2-n} a_{i+n}2^i \\ &= -a_{P-1}2^{P-1-n} + \sum_{i=0}^{P-2-n} a_{i+n}2^i.\end{aligned}$$

Using simple algebraic manipulation, (7), and (34), we can further rearrange the preceding equation as follows:

$$\begin{aligned}\operatorname{asr}(x,n) &= -a_{P-1}2^{P-1-n} + \sum_{i=-n}^{P-2-n} a_{i+n}2^i - \\ &\quad \sum_{i=-n}^{-1} a_{i+n}2^i \\ &= \tfrac{1}{2^n}(-a_{P-1}2^{P-1} + \sum_{i=0}^{P-2} a_i 2^i) - \tfrac{1}{2^n}\operatorname{mod}(x,2^n) \\ &= \tfrac{x-\operatorname{mod}(x,2^n)}{2^n} \\ &= \left\lfloor \tfrac{x}{2^n} \right\rfloor.\end{aligned}$$

Thus, the identity given in (32) holds. ■

Proposition 3: For all $x \in \mathbb{Z}$ and all $y \in \mathbb{Z}$ except $y = 0$, the following identity holds:

$$\lceil x/y \rceil = \lfloor (x+y-1)/y \rfloor. \tag{35}$$

Proof: One can easily confirm that the mod function satisfies the relationships

$$\operatorname{mod}(x,y) = x \quad \text{for } 0 \leq x \leq y-1 \text{ and } y \neq 0, \text{ and}$$

$$\operatorname{mod}(x+y,y) = \operatorname{mod}(x,y).$$

From the two preceding properties, we can deduce

$$\operatorname{mod}(-x,y) = y - 1 - \operatorname{mod}(x-1,y) \quad \text{for } y \neq 0. \tag{36}$$

Now we consider the expression on the right-hand side of equation (35). Using (7), we can write

$$\begin{aligned}\lfloor (x+y-1)/y \rfloor &= \lfloor (x-1)/y \rfloor + 1 \\ &= (x+y-1-\operatorname{mod}(x-1,y))/y.\end{aligned} \tag{37}$$

Using (8) and (36), we can further manipulate (37) as follows:

$$\lfloor (x+y-1)/y \rfloor = (x+\operatorname{mod}(-x,y))/y = \lceil x/y \rceil.$$

Thus, the identity given in (35) holds. ■

III. Rounding Error Derivations

Proposition 4: Consider the quantity $\frac{x}{2^F}$ where x is a random integer and F is a strictly positive integer constant. If we suppose that frac $\frac{x}{2^F}$ may assume all 2^F (distinct) possible values, then the error incurred by rounding the quantity $\frac{x}{2^F}$ using the the floor, biased floor, ceiling, biased ceiling, truncation, and biased truncation operators, is bounded as follows:

$$\begin{gathered}
-\tfrac{2^F-1}{2^F} \leq \left\lfloor\tfrac{x}{2^F}\right\rfloor - \tfrac{x}{2^F} \leq 0, \quad -\tfrac{2^{F-1}-1}{2^F} \leq \text{bfloor}\left(\tfrac{x}{2^F}\right) - \tfrac{x}{2^F} \leq \tfrac{1}{2}, \\
0 \leq \left\lceil\tfrac{x}{2^F}\right\rceil - \tfrac{x}{2^F} \leq \tfrac{2^F-1}{2^F}, \quad -\tfrac{1}{2} \leq \text{bceil}\left(\tfrac{x}{2^F}\right) - \tfrac{x}{2^F} \leq \tfrac{2^{F-1}-1}{2^F}, \\
-\tfrac{2^F-1}{2^F} \leq \text{trunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F} \leq \tfrac{2^F-1}{2^F}, \quad \text{and} \quad -\tfrac{1}{2} \leq \text{btrunc}\left(\tfrac{x}{2^F}\right) - \tfrac{x}{2^F} \leq \tfrac{1}{2}.
\end{gathered} \tag{38}$$

If we further suppose that all 2^F possible values of frac $\frac{x}{2^F}$ are equiprobable for $r < 0$ and are also equiprobable for $r \geq 0$, then rounding $\frac{x}{2^F}$ with the floor, biased floor, ceiling, biased ceiling, truncation, and biased truncation operators incurs the respective mean absolute errors:

$$\begin{gathered}
\mathrm{E}\left(\left|\left\lfloor\tfrac{x}{2^F}\right\rfloor - \tfrac{x}{2^F}\right|\right) = \tfrac{2^F-1}{2^{F+1}}, \quad \mathrm{E}\left(\left|\text{bfloor}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \tfrac{1}{4}, \\
\mathrm{E}\left(\left|\left\lceil\tfrac{x}{2^F}\right\rceil - \tfrac{x}{2^F}\right|\right) = \tfrac{2^F-1}{2^{F+1}}, \quad \mathrm{E}\left(\left|\text{bceil}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \tfrac{1}{4}, \\
\mathrm{E}\left(\left|\text{trunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \tfrac{2^F-1}{2^{F+1}}, \quad \text{and} \quad \mathrm{E}\left(\left|\text{btrunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \tfrac{1}{4}.
\end{gathered} \tag{39}$$

Proof: From (6) and (2), we can show that the rounding error incurred by the floor, biased floor, ceiling, and biased ceiling operators is, respectively, given by:

$$\left\lfloor\tfrac{x}{2^F}\right\rfloor - \tfrac{x}{2^F} = \tfrac{-\operatorname{mod}(x,2^F)}{2^F} \tag{40}$$

$$\text{bfloor}\left(\tfrac{x}{2^F}\right) - \tfrac{x}{2^F} = \tfrac{2^{F-1}-\operatorname{mod}(2^{F-1}+x,2^F)}{2^F} \tag{41}$$

$$\left\lceil\tfrac{x}{2^F}\right\rceil - \tfrac{x}{2^F} = \tfrac{\operatorname{mod}(-x,2^F)}{2^F} \tag{42}$$

$$\text{bceil}\left(\tfrac{x}{2^F}\right) - \tfrac{x}{2^F} = \tfrac{\operatorname{mod}(2^{F-1}-x,2^F)-2^{F-1}}{2^F} \tag{43}$$

Using the definitions of the trunc and btrunc functions given by (4) and (5), respectively, and the identities (40)–(43), we can easily deduce:

$$\text{trunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F} = \begin{cases} \tfrac{-\operatorname{mod}(x,2^F)}{2^F} & \text{for } x \geq 0 \\ \tfrac{\operatorname{mod}(-x,2^F)}{2^F} & \text{for } x < 0 \end{cases} \tag{44}$$

$$\text{btrunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F} = \begin{cases} \tfrac{2^{F-1}-\operatorname{mod}(2^{F-1}+x,2^F)}{2^F} & \text{for } x \geq 0 \\ \tfrac{\operatorname{mod}(2^{F-1}-x,2^F)-2^{F-1}}{2^F} & \text{for } x < 0 \end{cases} \tag{45}$$

Since $\operatorname{mod}(\pm x, 2^F)$ and $\operatorname{mod}(2^{F-1} \pm x, 2^F)$ can assume any integer value on $[0, 2^F - 1]$, we can easily deduce all of the error bounds in (38) from (40)–(45). For example, consider the case of the floor operator. The rounding error in this case is given by (40). Clearly, the minimum value of the rounding error is $-\frac{2^F-1}{2^F}$ (occurring when $\operatorname{mod}(x, 2^F) = 2^F - 1$). Similarly, the maximum value of the rounding error is 0 (occurring when $\operatorname{mod}(x, 2^F) = 0$). Thus, we have that $-\frac{2^F-1}{2^F} \le \lfloor \frac{x}{2^F} \rfloor - \frac{x}{2^F} \le 0$. The remaining error bounds in (38) can be shown to hold in a similar manner.

Now, let us consider the mean absolute error expressions for the various rounding operators. Since, by assumption, all 2^F possible values for frac $\frac{x}{2^F}$ are equiprobable, we know that $\operatorname{mod}(\pm x, 2^F)$ and $\operatorname{mod}(2^{F-1} \pm x, 2^F)$ are both uniformly distributed over the integer values on $[0, 2^F - 1]$. Consequently, in the case of the floor, biased floor, ceiling, and biased ceiling operators, the mean absolute error can be found by computing the expected value over the 2^F possible absolute error values (associated with the 2^F distinct fractional parts) using (40)–(45). Straightforward analysis and algebraic manipulation yield the following:

$$\mathrm{E}\left(\left|\lfloor \tfrac{x}{2^F} \rfloor - \tfrac{x}{2^F}\right|\right) = \mathrm{E}\left(\left|\tfrac{-\operatorname{mod}(x,2^F)}{2^F}\right|\right) = \tfrac{1}{2^F} \sum_{i=0}^{2^F-1} \left|\tfrac{-i}{2^F}\right| = \tfrac{2^F-1}{2^{F+1}} \tag{46}$$

$$\mathrm{E}\left(\left|\operatorname{bfloor}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \mathrm{E}\left(\left|\tfrac{2^{F-1}-\operatorname{mod}(2^{F-1}+x,2^F)}{2^F}\right|\right) = \tfrac{1}{2^F} \sum_{i=0}^{2^F-1} \left|\tfrac{2^{F-1}-i}{2^F}\right| = \tfrac{1}{4} \tag{47}$$

$$\mathrm{E}\left(\left|\lceil \tfrac{x}{2^F} \rceil - \tfrac{x}{2^F}\right|\right) = \mathrm{E}\left(\left|\tfrac{\operatorname{mod}(-x,2^F)}{2^F}\right|\right) = \tfrac{1}{2^F} \sum_{i=0}^{2^F-1} \left|\tfrac{i}{2^F}\right| = \tfrac{2^F-1}{2^{F+1}} \tag{48}$$

$$\mathrm{E}\left(\left|\operatorname{bceil}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \mathrm{E}\left(\left|\tfrac{\operatorname{mod}(2^{F-1}-x,2^F)-2^{F-1}}{2^F}\right|\right) = \tfrac{1}{2^F} \sum_{i=0}^{2^F-1} \left|\tfrac{i-2^{F-1}}{2^F}\right| = \tfrac{1}{4} \tag{49}$$

Let us now consider the mean absolute error for the trunc function. From (46), (48), and the definition of the trunc function in (4), it follows that

$$\mathrm{E}\left(\left|\operatorname{trunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \Pr(\tfrac{x}{2^F} < 0)\,\mathrm{E}\left(\left|\lceil \tfrac{x}{2^F} \rceil - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} < 0\right) + \Pr(\tfrac{x}{2^F} \ge 0)\,\mathrm{E}\left(\left|\lfloor \tfrac{x}{2^F} \rfloor - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} \ge 0\right) \tag{50}$$

Since, by assumption, all 2^F possible values of frac $\frac{x}{2^F}$ are equiprobable for $\frac{x}{2^F} < 0$, we have from (48)

$$\mathrm{E}\left(\left|\lceil \tfrac{x}{2^F} \rceil - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} < 0\right) = \tfrac{2^F-1}{2^{F+1}}. \tag{51}$$

Similarly, since all 2^F possible values of frac $\frac{x}{2^F}$ are assumed to be equiprobable for $\frac{x}{2^F} \ge 0$, we have from (46) that

$$\mathrm{E}\left(\left|\lfloor \tfrac{x}{2^F} \rfloor - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} \ge 0\right) = \tfrac{2^F-1}{2^{F+1}}. \tag{52}$$

By substituting (51) and (52) into (50) and then observing that $\Pr(\frac{x}{2^F} < 0) + \Pr(\frac{x}{2^F} \geq 0) = 1$, we obtain

$$\mathrm{E}\left(\left|\mathrm{trunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \Pr(\tfrac{x}{2^F} < 0)\left(\tfrac{2^F-1}{2^{F+1}}\right) + \Pr(\tfrac{x}{2^F} \geq 0)\left(\tfrac{2^F-1}{2^{F+1}}\right) = \tfrac{2^F-1}{2^{F+1}}.$$

Therefore, the identity in (39) for the truncation operator holds.

Consider now the mean absolute error for the biased truncation operator. From (47), (49), and the definition of the btrunc function in (5), it follows that

$$\begin{aligned}\mathrm{E}\left(\left|\mathrm{btrunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = &\Pr(\tfrac{x}{2^F} < 0)\,\mathrm{E}\left(\left|\mathrm{bceil}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} < 0\right) \\ &+ \Pr(\tfrac{x}{2^F} \geq 0)\,\mathrm{E}\left(\left|\mathrm{bfloor}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} \geq 0\right)\end{aligned} \tag{53}$$

Since, by assumption, all 2^F possible values of $\mathrm{frac}\,\frac{x}{2^F}$ are equiprobable for $\frac{x}{2^F} < 0$, we have from (49) that

$$\mathrm{E}\left(\left|\mathrm{bceil}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} < 0\right) = \tfrac{1}{4} \tag{54}$$

Similarly, since all 2^F possible values of $\mathrm{frac}\,\frac{x}{2^F}$ are equiprobable for $\frac{x}{2^F} \geq 0$, we have from (47) that

$$\mathrm{E}\left(\left|\mathrm{bfloor}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right| \mid \tfrac{x}{2^F} \geq 0\right) = \tfrac{1}{4} \tag{55}$$

By substituting (54) and (55) into (53) and observing that $\Pr(\frac{x}{2^F} < 0) + \Pr(\frac{x}{2^F} \geq 0) = 1$, we obtain

$$\mathrm{E}\left(\left|\mathrm{btrunc}(\tfrac{x}{2^F}) - \tfrac{x}{2^F}\right|\right) = \tfrac{1}{4}\Pr(\tfrac{x}{2^F} < 0) + \tfrac{1}{4}\Pr(\tfrac{x}{2^F} \geq 0) = \tfrac{1}{4} \tag{56}$$

Therefore, the identity in (39) for the biased truncation operator holds. Thus, we have shown that all of the identities in (39) hold.

■

ACKNOWLEDGMENT

The authors would like to thank Dr. Hongjian Shi and the anonymous reviewers for their useful comments and suggestions.

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] H. Chao, P. Fisher, and Z. Hua, "An approach to integer wavelet transforms for lossless for image compression," in *Proc. of International Symposium on Computational Mathematics*, Guangzhou, China, Aug. 1997, pp. 19–38.

[6] P. Lux, "A novel set of closed orthogonal functions for picture coding," *Archiv fur Elektronik und Uebertragungstechnik*, vol. 31, no. 7, pp. 267–274, July 1977.

[7] F. A. M. L. Bruekers and A. W. M. van den Enden, "New networks for perfect inversion and perfect reconstruction," *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 1, pp. 130–137, Jan. 1992.

[8] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, no. 2, pp. 186–200, 1996.

[9] R. S. Garfinkel and G. L. Nemhauser, *Integer Programming*, John Wiley and Sons, Toronto, 1972.

[10] P. Hao and Q. Shi, "Matrix factorizations for reversible integer mapping," *IEEE Trans. on Signal Processing*, vol. 49, no. 10, pp. 2314–2324, Oct. 2001.

[11] J. E. Shockley, *Introduction to Number Theory*, Holt, Rinehart, and Winston, Toronto, 1967.

[12] ISO/IEC, *ISO/IEC 15444-1, Information technology — JPEG 2000 image coding system — Part 1: Core coding system*, 2001.

[13] 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.

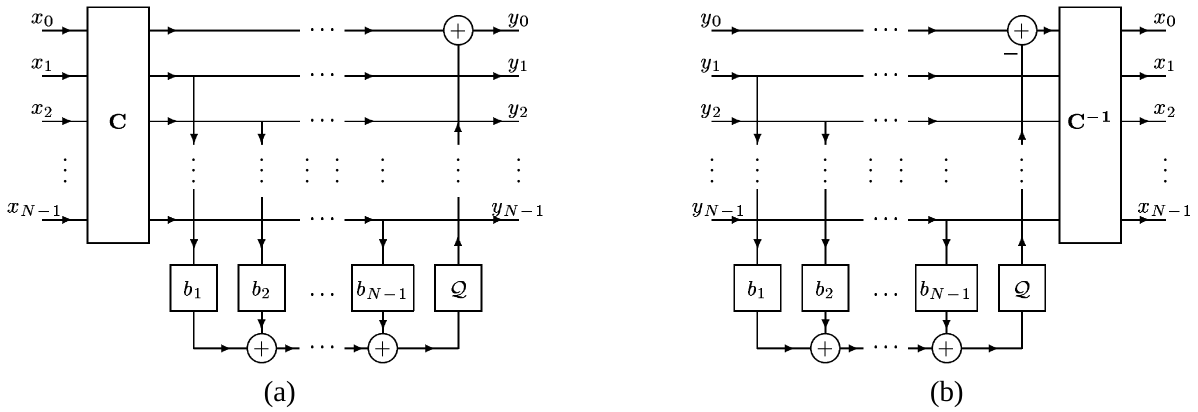

Fig. 1. Network realization of the generalized S transform. (a) Forward transform and (b) inverse transform.

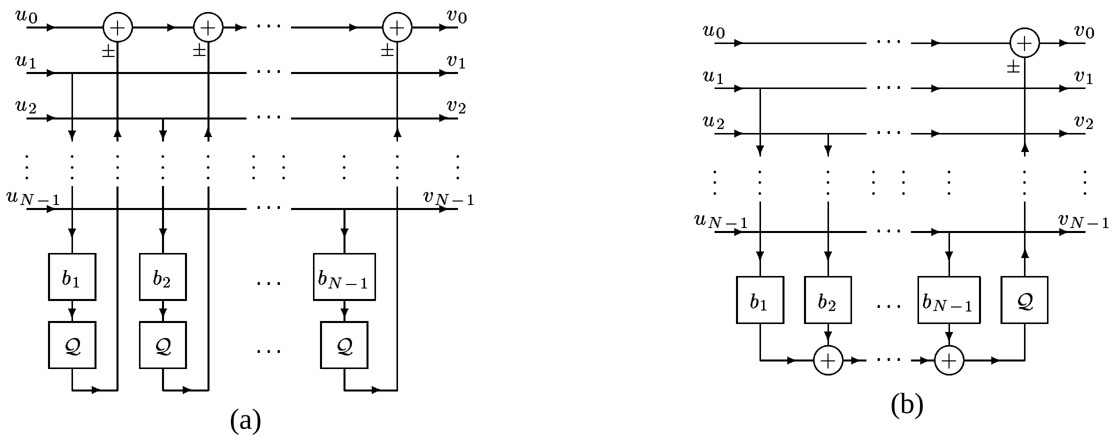

Fig. 2. Network for a group of ladder steps having the same output channel with rounding performed for (a) each ladder step separately, and (b) all of the ladder steps together.

TABLE I

Error characteristics for various rounding operators (for $F \geq 1$)

Operator	Error Interval		PAE		MAE	
	Exact	For Large F	Exact	For Large F	Exact	For Large F
floor	$[-\frac{2^F-1}{2^F}, 0]$	$(-1, 0]$	$\frac{2^F-1}{2^F}$	1	$\frac{2^F-1}{2^{F+1}}$	$\frac{1}{2}$
bfloor	$[-\frac{2^{F-1}-1}{2^F}, \frac{1}{2}]$	$(-\frac{1}{2}, \frac{1}{2}]$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$
ceiling	$[0, \frac{2^F-1}{2^F}]$	$[0, 1)$	$\frac{2^F-1}{2^F}$	1	$\frac{2^F-1}{2^{F+1}}$	$\frac{1}{2}$
bceil	$[-\frac{1}{2}, \frac{2^{F-1}-1}{2^F}]$	$[-\frac{1}{2}, \frac{1}{2})$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$
trunc	$[-\frac{2^F-1}{2^F}, \frac{2^F-1}{2^F}]$	$(-1, 1)$	$\frac{2^F-1}{2^F}$	1	$\frac{2^F-1}{2^{F+1}}$	$\frac{1}{2}$
btrunc	$[-\frac{1}{2}, \frac{1}{2}]$	$[-\frac{1}{2}, \frac{1}{2}]$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$

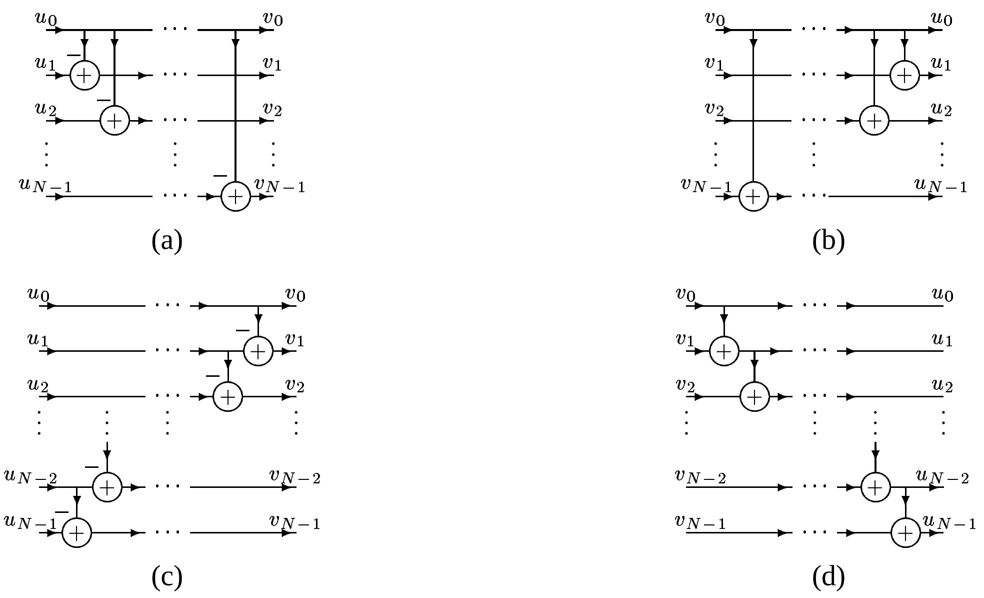

Fig. 3. Networks for realizing particular forms of the $\mathbf{C}$ and $\mathbf{C}^{-1}$ matrices. The networks for type-1 systems: (a) forward and (b) inverse networks. The networks for type-2 systems: (c) forward and (d) inverse networks.

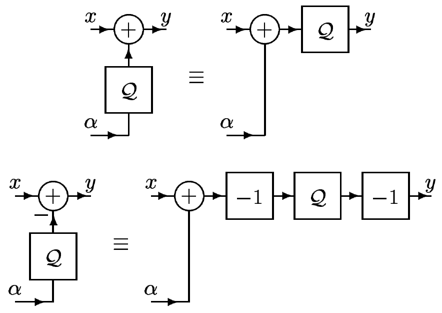

Fig. 4. Transformations for integer-shift invariant rounding operator.

TABLE II

DEVIATION FROM THE TRUE LUMINANCE FOR THE MRCT AND RCT

	MRCT		RCT	
Image	MAE	PAE	MAE	PAE
`flowers`	0.654	1.635	6.515	25.326
`food`	0.734	1.777	8.860	34.821
`sailboat`	0.566	1.438	1.888	20.580
`tile`	0.776	1.369	9.591	14.560