EFFICIENT COMPUTATION OF THE DISCRETE PASCAL TRANSFORM

Athanassios Skodras

Computer Science, Hellenic Open University
Tsamadou 13-15, GR-26222 Patras, Greece
phone: + (30) 2610 367522, fax: + (30) 2610 367520, email: skodras@eap.gr
web: dsmc.eap.gr

ABSTRACT

The recently proposed discrete Pascal transform possesses a computational complexity for an N-point vector of the order of N^2 for both multiplications and additions. In the present work an efficient structure is proposed, which eliminates the multiplications and halves the number of additions.

1. INTRODUCTION

The discrete Pascal transform (DPT) was recently introduced by Aburdene and Goodman [1]. It belongs to the family of the discrete polynomial transforms, and it is based on a variation of the Pascal matrix [1,2,3]. Such transforms are used in signal and image processing for communication and control systems [1,4,5]. Two interesting applications that have recently reported are those of filter design and discrete-time signal interpolation [6,7].

Pascal's triangle is a geometric arrangement of the binomial coefficients in a triangle [2, 3]. The first eight rows of Pascal's triangle are shown in Fig. 1.

```
                    1
                1       1
            1       2       1
        1       3       3       1
    1       4       6       4       1
1       5      10      10       5       1
    6      15      20      15       6       1
1       7      21      35      35      21       7       1
```

(a)

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
1  5  10 10 5  1
1  6  15 20 15 6  1
1  7  21 35 35 21 7  1
```

(b)

Figure 1 - The first eight rows of Pascal's triangle in two different presentation variations

Each number of the triangle is equal to the binomial coefficient $a_{ij,}$ where

$$a_{ij} \equiv \binom{i}{j} \equiv \frac{i!}{j!(i-j)!} \tag{1}$$

with *i, j* non-negative integers, $i \geq j$ and $\binom{i}{0} = \binom{i}{i} = 1$.

As with all discrete transforms the main concern is their increased computational complexity, which by their definition as matrix multiplications, is of the order of N^2 for data sequences of length *N*. In the present work a reduced computational complexity scheme is proposed for the DPT. This scheme has a complexity of *½ N(N-1)* additions and no multiplications. Such a reduced complexity will facilitate the use of the DPT in more applications.

The work is structured as follows: In section 2 the DPT is introduced. Its complexity is analyzed in section 3. A computationally efficient realization of the DPT is given in section 4.

2. THE DISCRETE PASCAL TRANSFORM

The discrete Pascal transform (DPT) **X** of the 1D data vector **x** is defined as

$$\mathbf{X} = \mathbf{P}\,\mathbf{x} \tag{2}$$

where **x**, **X** are *N×1* vectors and **P** is the *N×N* Pascal transform matrix. The elements of the Pascal transform matrix are equal to

$$p_{ij} = \begin{cases} (-1)^j a_{ij} & \text{for } i \geq j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

with *i*=0, 1, 2,…, *N*-1.

From (3) it is easily deduced that the elements of the Pascal transform matrix are obtained directly from the Pascal's triangle entries (Fig. 1b) by alternating the signs of the columns. The Pascal's transform matrices for *N*=2, *N*=4, and *N*=8 are given below. (Zero elements have been denoted by dots for presentation clarity purposes).

$$\mathrm{P}_2 = \begin{bmatrix} 1 & \cdot \\ 1 & -1 \end{bmatrix} \tag{4}$$

$$\mathrm{P}_4 = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ 1 & -1 & \cdot & \cdot \\ 1 & -2 & 1 & \cdot \\ 1 & -3 & 3 & -1 \end{bmatrix} \tag{5}$$

$$\mathrm{P}_8 = \begin{bmatrix} 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & -1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & -2 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & -3 & 3 & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & -4 & 6 & -4 & 1 & \cdot & \cdot & \cdot \\ 1 & -5 & 10 & -10 & 5 & -1 & \cdot & \cdot \\ 1 & -6 & 15 & -20 & 15 & -6 & 1 & \cdot \\ 1 & -7 & 21 & -35 & 35 & -21 & 7 & -1 \end{bmatrix} \tag{6}$$

The basic properties of the **P** matrices are the following:

- The elements of the first column are equal to 1.
- All matrices are lower triangular.
- The sum of the elements of each row (except of the first one) is equal to zero.
- All matrices are equal to their inverses.

The DPT of eq. (2) can also be expressed as:

$$X_k = \sum_{n=0}^{N-1} p_{kn} x_{n,} \quad n=0, 1, \ldots, N\text{-}1 \tag{7}$$

where x_n is the data sequence, X_k are the transform coefficients and k=0, 1, …, N-1.

Since the elements of the Pascal transform matrix that are above the diagonal are equal to zero, i.e. p_{kn} =0 for $k>n$, the upper limit of the summation of eq. (7) becomes equal to k (instead of N-1). Thus the forward DPT can be rewritten as follows:

$$X_k = \sum_{n=0}^{k} p_{kn} x_n, \quad n=0, 1, \ldots, N\text{-}1 \tag{8}$$

or

$$X_k = \sum_{n=0}^{k} (-1)^n \binom{k}{n} x_n \tag{9}$$

It was mentioned above that the Pascal transform matrix is equal to its inverse,

$$\mathbf{P^{-1} = P} \tag{10}$$

Thus, the inverse DPT is written in matrix form as

$$\mathbf{x = P\,X} \tag{11}$$

This can also be expressed as

$$x_n = \sum_{k=0}^{n} p_{nk} X_k, \qquad k = 0,1,\ldots, N-1 \tag{12}$$

or

$$x_n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} X_k, \qquad k = 0,1,\ldots, N-1 \tag{13}$$

Equations (9) and (13) constitute the forward and inverse DPT, respectively.

The basis functions of the DPT are calculated via the polynomials $P(x, j)$, where

$$\begin{aligned} P(x, j) &= p_{x\,j} = (-1)^j a_{x\,j} = (-1)^j \binom{x}{j} \\ &= (-1)^j \frac{x(x-1)(x-2)\ldots(x-j+1)}{j!} \end{aligned} \tag{14}$$

for $j \geq x \geq 0$. In general, each polynomial $P(x, j+1)$ can be recursively computed by means of the formula

$$P(x, j+1) = -\frac{1}{j+1}(x-j)\, P(x, j) \tag{15}$$

These polynomials when evaluated at the discrete points $x = 0,1,\ldots, N-1$ produce the columns of the Pascal transform matrix $\mathrm{P_N}$, which actually constitute the basis vectors of the transform.

3. COMPUATIONAL COMPLEXITY

The computation of the DPT of an N-point vector based on matrix definition of eq. (2) results in N^2 multiplications and $N(N\text{-}1)$ additions, i.e.

$$M_N = N^2 \tag{16}$$

and

$$A_N = N(N\text{-}1) \tag{17}$$

If we take into account that the upper triangle entries of the transform matrix are zero and the diagonal entries are equal to 1 or -1, i.e. no multiplications are needed, then the operations counts become:

$$M_N = \frac{N^2 - N}{2} = \frac{1}{2} N(N-1) \tag{18}$$

$$A_N = 1 + 2 + 3 + \ldots + (N-1) = \frac{1}{2} N(N-1) \tag{19}$$

This is also the case when eq. (9) is used for the computation of the DPT.

Going one step further and taking into account that the entries of the first column of the matrix are all equal to 1, i.e. no multiplications are needed, M_N reduces to

$$M_N = \frac{1}{2} N(N-1) - (N-1) = \frac{1}{2} N(N-3) + 1 \tag{20}$$

4. EFFICIENT COMPUTATION

The computational complexity of the DPT could be highly reduced if the computations were performed in a recursive

manner, namely if the transform coefficients X_0, X_1,…, X_{k-1} were used for the computation of X_k [8].

4.1 Computation of a 2-point DPT (*N*=2)

Let $x = [x_0 x_1]^T$ be the data and $X = [X_0 X_1]^T$ be the DPT coefficients. The transform coefficients $\boldsymbol{X}$ are calculated through eq. (2) using the transform matrix (4), as

$$\mathbf{X = P\,x} \Rightarrow \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} \Rightarrow \begin{cases} X_0 = x_0 \\ X_1 = x_0 - x_1 \end{cases} \tag{21}$$

The flow graph for the calculation of (21) is shown in Fig. 2. In fact, only the lower part of the known FFT butterfly is used, where we have only a subtraction of the two inputs. The complexity of the calculation is equal to: M_2=0, A_2=1.

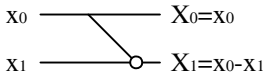

Figure 2 - The 2-point DPT butterfly

4.2 Calculation of a 4-point DPT (*N*=4)

For the calculation of the 4-point DPT we have:

$$\Rightarrow \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -3 & 3 & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} =$$

$$\mathbf{X = P\,x} \begin{cases} X_0 = x_0 \\ X_1 = x_0 - x_1 \\ X_2 = (x_0 - x_1) - (x_1 - x_2) \\ X_3 = x_0 - 3x_1 + 3x_2 - x_3 = \\ (x_0 - x_1) - 2(x_1 - x_2) + (x_2 - x_3) \\ = [(x_0 - x_1) - (x_1 - x_2)] - (x_1 - x_2) + (x_2 - x_3) \\ = X_2 - [(x_1 - x_2) - (x_2 - x_3)] \end{cases} \tag{22}$$

In fact we have decomposed the 4×4 Pascal transform matrix P (or P_4 for clarity) into a product of simple lower triangular matrices Q, the entries of which consist only of 0, 1, and -1, namely $P_4=Q_3Q_2Q_1$, where

$$Q_3 = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & 1 & -1 \end{bmatrix}$$

$$Q_2 = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & 1 & -1 & \cdot \\ \cdot & \cdot & 1 & -1 \end{bmatrix}$$

$$Q_1 = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ 1 & -1 & \cdot & \cdot \\ \cdot & 1 & -1 & \cdot \\ \cdot & \cdot & 1 & -1 \end{bmatrix}$$

In general, the entries $q_{i,j}$ of each matrix Q_k are defined as:

$$Q_k = [q_{ij}]_k = \begin{cases} 1 & \text{for } i = j < k \text{ or } i \geq k \ \wedge \ i = j+1 \\ -1 & \text{for } i = j \geq k \\ 0 & \text{otherwise} \end{cases}$$

The flow graph for the calculation of the 4-point DPT according to (22) is shown in Fig. 3. The calculation is completed in three stages (*N*-1= 3) and only six subtractions (3+2+1=6) are required, or M_4=0 and A_4=6.

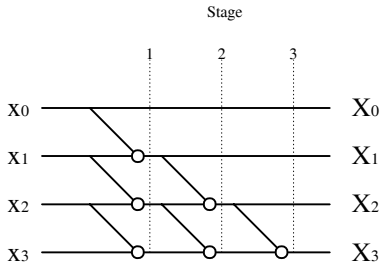

Figure 3 - The 4-point DPT flow graph

In a similar way the 8-point DPT can be deduced, as depicted in Fig. 4.

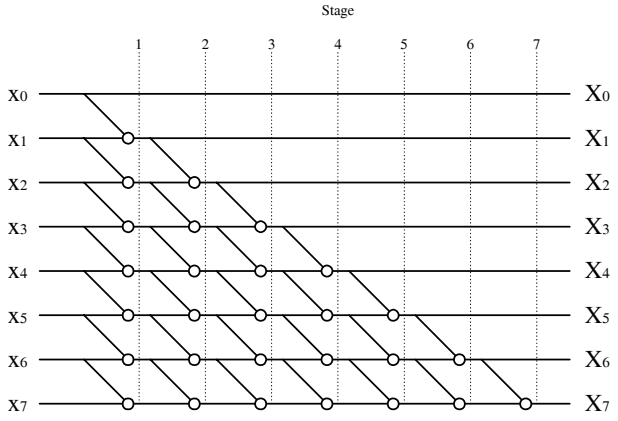

Figure 4. The 8-point DPT flow graph

4.3 Calculation of an *N*-point DPT

In the flow graph of an *N*-point DPT there would exist *N-1* stages, and: *N-1* butterflies at stage *1*, *N-2* butterflies at stage *2*,…,*N-m* butterflies at stage *m*. For each butterfly only one subtraction is needed. No multiplications are needed for the whole computation. The number of subtractions is equal to 1 for stage *N-1*, 2 for stage *N-2*,...,*N-1* for stage 1. This gives a total of

$$1+2+3+...+N-1=\sum_{m=1}^{N-1} m = N\frac{N-1}{2} \qquad (23)$$

subtractions. Thus the total computational complexity for the N-point DPT is

$$M_N = 0 \qquad (24)$$

and

$$A_N = \frac{1}{2}N(N-1) \qquad (25)$$

The computational complexity for each of the methods presented above are summarized in Table I. It is clear that the proposed approach, which is based on a butterfly structure, possesses the lowest complexity and actually constitutes a fast Pascal transform (FPT).

Table I - Computational complexity for the 1D DPT

	Direct matrix multiplication (eq. 16,17)	Lower triangular matrix (eq. 18,19)	Lower triangular with 1's in the first column (eq. 20)	Proposed FPT (butterfly based) (eq. 24,25)
M_N	N^2	½ $N(N$-1)	½ $N(N$-1)-(N-1)	0
A_N	$N(N$-1)	½ $N(N$-1)	½ $N(N$-1)	½ $N(N$-1)

5. CONCLUSIONS

The discrete Pascal transform is a new promising tool for many signal and image processing applications. The triangular structure of the discrete Pascal transform matrix produces a useful localization property, something that makes it attractive in bump and edge detection applications [1]. The computation burden for the 1D case is always high and of the order of N^2 for both multiplications and additions. In the present work a new approach has been proposed, which eliminates the multiplications and halves the number of additions. The calculation of every new transform coefficient is achieved by pairing the additions at each stage and combining the results of the previous stage.

6. ACKNOWLEDGEMENTS

The insightful remarks, on the DPT and its connection to high pass filtering, by Professor A.G. Constantinides, DEE, Imperial College, London, are gratefully acknowledged. This work was funded by the European Social Fund (Programme 'Pythagoras': contract no. 89188).

REFERENCES

[1] M.F. Aburdene and T.J. Goodman: "The Discrete Pascal Transform", *IEEE Signal Processing Letters*, Vol. 12, No. 7, pp. 493-495, July 2005.

[2] Pascal's Triangle – Wikipedia: [on-line] http://en.wikipedia.org/wiki/Pascal's_triangle

[3] Pascal's Triangle – Mathworld: [on-line] http://matworld.wolfram.com/PascalsTriangle.html

[4] G. Mandyam and N. Ahmed: "The Discrete Laguerre Transform: Derivation and Applications", *IEEE Trans on Signal Processing*, Vol. 44, No. 12, pp. 2925-2931, Dec. 1996.

[5] S.A.M. Gilani and A.N. Skodras: "Watermarking of Images in the DLT Domain", *Technical Report TR2000-03-02*, Computer Technology Institute, Patras, Greece, March 2000.

[6] M. Aburdene and T. Goodman: "Discrete Polynomials and Filter Design", *2005 Conf. on Information Sciences and Systems*, The John Hopkins University, March 16-18, 2005.

[7] T.J. Goodman and M.F. Aburdene: "Interpolation Using the Discrete Pascal Transform", *2006 Conf. on Information Sciences and Systems*, Princeton University, March 22-24, 2006.

[8] A.N. Skodras: "On the Computation of the Discrete Pascal Transform", *Technical Report HOU-CS-TR-2005-06-EN*, Hellenic Open University, Patras, Greece, Dec. 2005.