CE 7 : Mode-dependent DCT/DST without 4*4 multiplication for intra prediction
JCTVC-E125

Ankur Saxena & Felix C. Fernandes
Samsung Electronics, Co. Ltd.
Dallas R&D Center

Block Diagram for Unified Intra Prediction and Transform

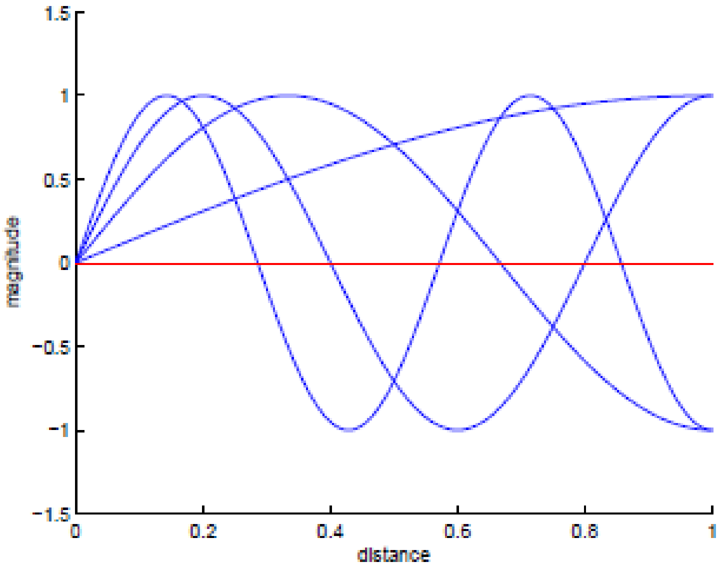
- **Current:** DCT is used as the transform for all block sizes 4*4, 8*8, 16*16 and 32*32 following intra-prediction

- ***<u>Proposed</u>*** : Mode-dependent DCT/DST for intra prediction at sizes 4*4 and 8*8

Mode-Dependent DCT/DST

- Depending on intra prediction mode, the transform used is DCT or DST Type-7

$$[T_S]_{j,i} = \left(\frac{2}{\sqrt{2N+1}} \sin \frac{(2j-1)i\pi}{2N+1} \right)$$



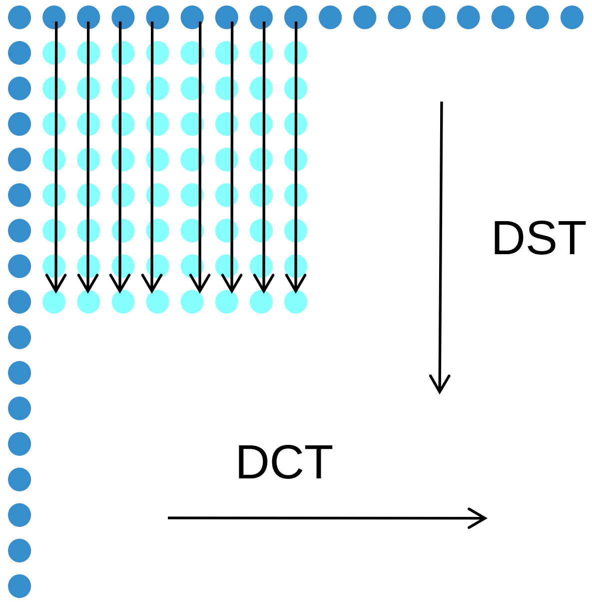
- Derivation and details in
  Han, Saxena & Rose (ICASSP 2010) and Saxena & Fernandes (JCTVC-C108, JCTVC-D033)

Sine bases

Mapping from Intra Prediction Modes to DCT or DST

	Mode	Vertical Transform	Horizontal Transform
0	VER to VER + 8	DST	DCT
1	HOR to HOR + 8	DCT	DST
2	DC	DCT	DCT
3	VER-8 to VER – 1	DST	DST
4	HOR-8 to HOR-1	DST	DST

Example : Vertical Mode



- Vertical Mode

- DST in vertical direction
- DCT in horizontal direction

- Implementation in HM 2.0

Implementation in HM 2.0

- Experiments were performed for Intra HE, Intra LC, Random Access HE, Random Access LC configurations according to stipulated conditions in CE 7 description (JCTVC-D607, Robert Cohen et al.)

- **Scanning :** Uses HM 2.0 scanning for the transform coefficients ***without any modification***

- **DCT Matrices** Uses HM 2.0 core DCT matrices ***without any modification***

- **Quantization & Inverse Quantization Tables:** Use HM 2.0 quantization tables ***without any modification***

- ***DST Matrices :*** Use DST-Type 7 matrices (norm-corrected) and do not make any changes to the DST transform matrix elements

4*4 DST Matrices

Forward 4*4 DST matrix

{29,	117,	84,	87}
{55,	117,	-29,	-133}
{74,	0,	-74,	117}
{84,	-117,	55,	-46}

Inverse 4*4 DST matrix

{	29,	58,	84,	43}
{	55,	58,	-29,	-66}
{	74,	0,	-74,	58}
{	84,	-58,	55,	-23}

- **Basis Vectors along columns**

DST Matrices: *In our proposal, we use the norm-corrected DST matrices without any further modification*

The elements in 1st and 3rd basis vectors are same (upto sign and permutation)

Fast 4*4 Forward DST

- The forward 4*4 DST matrix with input [x(0),x(1),x(2), x(3)] and output [y(0),y(1),y(2), y(3)] is implemented as:

c[0] = x[0] + x[3];
c[1] = x[1] + x[3];
c[2] = 74* x[2];

y[0] = 29* c[0] + 55 * c[1] + c[2]
y[1] = 117 * x[0] + x[1] - x[3])
y[2] = 84* c[0] - 29 * c [1] - c[2]
y[3] = 87* c[0] - 133 * c[1] + 117* x[2]

- 9 multiplications and 10 additions instead of 16 mults and 12 adds in full matrix multiplication.

Fast 4*4 Inverse DST

- Similarly, the inverse DST with input [y(0),y(1),y(2), y(3)] and reconstructed output [$x_r(0)$,$x_r(1)$,$x_r(2)$, $x_r(3)$] is implemented as:

$$c0 = y[0] + y[2];$$
$$c1 = 58 * y[1];$$
$$c2 = y[2] + y[3];$$

$$x_r[0] = 29 * c0 + c1 + 55 * c2 - ((y[3] \ll 3) + (y[3] \ll 2))$$
$$x_r[1] = 55 * c0 + c1 - 84 * c2 + (y[3] \ll 4) + (y[3] \ll 1)$$
$$x_r[2] = 74 * (y[0] - y[2]) + 58*y[3]$$
$$x_r[3] = 84 * c0 - c1 - 29 * c2 + (y[3] \ll 2) + (y[3] \ll 1)$$

- 9 multiplications, 16 adds, and 6 shifts instead of 16 mults and 12 adds in full matrix multiplication

Storage Requirement : Inverse 4*4 DST

- The inverse DST with input [y(0),y(1),y(2), y(3)] and reconstructed output [x_r(0),x_r(1),x_r(2), x_r(3)] is implemented as:

c0 = y[0] + y[2];

c1 = 58 * y[1];

c2 = y[2] + y[3];

x_r[0] = 29 * c0 + c1 + 55 * c2 - ((y[3]<<3) + (y[3]<<2))

x_r [1] = 55 * c0 + c1 - 84 * c2 + (y[3]<<4) + (y[3]<<1)

x_r [2] = 74 * (y[0] - y[2]) + 58*y[3]

x_r[3] = 84 * c0 - c1 -29 *c2 + (y[3]<<2) + (y[3]<<1)

- **Only 5 distinct elements {29,55,58,74,84}** required for implementing inverse 4*4 DST.

Storage Requirement : Forward 4*4 DST

- The forward 4*4 DST matrix with input [x(0),x(1),x(2), x(3)] and output [y(0),y(1),y(2), y(3)] is implemented as:

c[0] = x[0] + x[3];
c[1] = x[1] + x[3];
c[2] = 74* x[2];

y[0] = 29* c[0] + 55 * c[1] + c[2]
y[1] = 117 * x[0] + x[1] - x[3])
y[2] = 84* c[0] - 29 * c [1] - c[2]
y[3] = 87* c[0] - 133 * c[1] + 117* x[2]

Only 3 additional distinct elements {87,117,133} are required for implementing forward 4*4 DST.

Inverse 4*4 Fast DST: Elemental Adds and Shifts for Hardware Implementation

		Adds	Shifts
29	= 32 – 2 -1	2	2
55	= 64 – 8 -1	2	2
58	= 64 – 4 -2	2	3
74	= 64 + 8 +2	2	3
84	= 64 + 16+4	2	3
Total		10	13

Comments on Obtaining Fast 4*4 DST

- **We have not made any changes to the coefficients in DST matrix.**
  - **Especially to 1st and 3rd basis vectors which have same norm**

- Number of adds and shifts can be reduced slightly by changing some coefficients in the DST matrix
  - But this **reduces compression efficiency**
  - Also the **storage will increase** from 5 elements to more
  - Norms of basis vectors will change

- e.g., **Inverse DST matrix (basis vectors in columns):**

{ 29,	58,	84,	43}
{ **55**,	58,	-29,	-66}
{ 74,	0,	-74,	58}
{ 84,	-58,	**55**,	-23}

Original DST Matrix

{ 29,	58,	84,	43}
{ **55**,	58,	-29,	-66}
{ 74,	0,	-74,	58}
{ 84,	-58,	**54**,	-23}

Example of a Modified DST Matrix

Experiments for Mode-Dependent DCT/DST

	Anchor	Proposed Scheme	Comments	Cross-Checkers
1.	HM 2.0	DCT/DST at size 4*4 and 8*8 ; DCT at size 16 and 32	Fast 4*4 DST ; 8*8 DST using matrix multiplication	I2R (JCTVC-E099)
2.	HM 2.0	DCT/DST only at block size 4*4; DCT at size 8, 16 and 32	Fast 4*4 DST	DoCoMo(JCTVC-E460)

DCT/DST at 4*4 and 8*8. (Uses Fast 4*4 DST)

	Intra			Intra LoCo		
	Y BD-rate	U BD-rate	V BD-rate	Y BD-rate	U BD-rate	V BD-rate
Class A	-1.2	-1.7	-1.4	-2.1	-0.4	-0.2
Class B	-0.7	-1.2	-1.2	-1.3	-0.9	-0.8
Class C	-0.9	-1.1	-1.1	-1.7	-1.1	-1.1
Class D	-1.0	-1.0	-1.0	-1.7	-1.0	-1.0
Class E	-1.2	-2.0	-1.8	-2.3	-0.7	-1.0
All	-1.0	-1.4	-1.3	-1.8	-0.8	-0.8
Enc Time[%]	102%			100%		
Dec Time[%]	100%			103%		

	Random access			Random access LoCo		
	Y BD-rate	U BD-rate	V BD-rate	Y BD-rate	U BD-rate	V BD-rate
Class A	-0.4	-0.5	-0.7	-0.7	0.0	0.2
Class B	-0.3	-0.3	-0.2	-0.6	-0.1	-0.2
Class C	-0.5	-0.6	-0.5	-0.8	-0.5	-0.3
Class D	-0.4	-0.4	-0.6	-0.7	-0.2	-0.2
Class E						
All	-0.4	-0.5	-0.5	-0.7	-0.2	-0.1
Enc Time[%]	101%			104%		
Dec Time[%]	102%			103%		

DCT/DST at 4*4 only. (Fast 4*4 DST)

	Intra			Intra LoCo		
	Y BD-rate	U BD-rate	V BD-rate	Y BD-rate	U BD-rate	V BD-rate
Class A	-0.7	-0.7	-0.6	-1.1	-0.1	0.0
Class B	-0.6	-0.6	-0.6	-1.0	-0.3	-0.2
Class C	-0.8	-0.7	-0.6	-1.3	-0.5	-0.4
Class D	-0.9	-0.7	-0.7	-1.4	-0.5	-0.5
Class E	-1.0	-1.1	-1.1	-1.6	-0.3	-0.5
All	-0.8	-0.7	-0.7	-1.3	-0.3	-0.3
Enc Time[%]	102%			102%		
Dec Time[%]	101%			100%		

	Random access			Random access LoCo		
	Y BD-rate	U BD-rate	V BD-rate	Y BD-rate	U BD-rate	V BD-rate
Class A	-0.2	0.1	-0.4	-0.4	0.1	0.2
Class B	-0.3	-0.1	-0.2	-0.5	0.1	0.1
Class C	-0.5	-0.2	-0.2	-0.6	-0.2	-0.2
Class D	-0.4	-0.4	-0.4	-0.6	-0.1	-0.4
Class E						
All	-0.3	-0.2	-0.3	-0.5	0.0	-0.1
Enc Time[%]	103%			104%		
Dec Time[%]	100%			100%		

Summary (Fast DST at 4*4)

	Proposed Scheme	Gains : Intra HE, LC, Random Access HE, Random Access LC
1.	DCT/DST at size 4*4 and 8*8	**-1.0 %, -1.8 %, - 0.4 %, -0.7 %**
2.	DCT/DST only at block size 4*4	**-0.8 %, -1.3 %, -0.3 %, -0.5 %**

- Additional encoding/decoding complexity is negligible
- Combination of DCT/DST as a primary transform (at sizes 4*4 and 8*8) with a secondary transform improves compression efficiency.
  - JCTVC-E380 by Samsung discusses combining DCT/DST and Rotational Transform (ROT)

Complexity Evaluation

Evaluation Metrics			Core HM transform	Our Proposal
4x4	Average Transform Arithmetic Operations (ops/pixel)	Mults	0.0	2.3
		Adds	5.0	7.5
		Shifts	2.0	3.5
	Worst-Case Transform Arithmetic Operations (ops/pixel)	Mults	0.0	4.5
		Adds	5.0	10.0
		Shifts	2.0	5.0
	Bit Precision	Max. required (bits)	32	32
	Memory requirements	Additional for transform (bytes)	0	5
8x8	Average Transform Arithmetic Operations (ops/pixel)	Mults	0.0	8.0
		Adds	10.0	13.0
		Shifts	5.5	3.75
	Worst-Case Transform Arithmetic Operations (ops/pixel)	Mults	0.0	16.0
		Adds	10.0	16.0
		Shifts	5.5	5.5
	Bit Precision	Max. required (bits)	32	32
	Memory requirements	Additional for transform (bytes)	0	64

Conclusions

- Proposed mode-dependent DCT/DST as a transform for Intra coding
- No changes to DST matrix coefficients
  - To retain better compression efficiency
  - Less storage
- Fast DST at size 4*4
- **Recommend to adopt mode-dependent DCT/DST at block sizes 4*4 and 8*8 as the transform for Intra coding in HM.**

Thank You