IV. The Wei 16-State 4-D Code

The AT&T and Codex proposals for V.fast both suggest using a version of the 16-state 4-D code invented by L.F. Wei (L.-F. Wei, "Trellis-Coded Modulation with Multidimensional Constellations," *IEEE Trans. Information Theory*, vol. IT-33, July 1987, pp. 327-338). This code has a fundamental coding gain of 4.52 dB and an effective coding gain of 4.20 dB when nearest neighbors are taken into account. (See G.D. Forney, "Coset Codes I: Introduction and Geometrical Classification," IEEE Trans. on Information Theory, Vol. IT-34, Sept. 1988, pp. 1123-1151.) Its effective coding gain is about 0.7 dB more than that of the 8-state 2-D V.32bis code and has about the same computational decoding complexity.

A. Forming the 4-D Constellation

The 4-D constellation is formed by partitioning the translated 4-D lattice $Z^4 + (\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2})$ into 8 translated 4-D cosets. The 4-D points are transmitted by sequentially sending a pair of 2-D symbols using QAM modulation. The partitioning is performed in two steps. First, the 2-D constellation $Z^2 + (\frac{1}{2},\frac{1}{2})$ is partitioned into the four subsets A, B, C, and D shown in Fig. IV-1. $A - (\frac{1}{2},\frac{1}{2})$ is the 2-D lattice $2Z^2$ and B, C, and D correspond to 2-D cosets of $2Z^2$ in Z^2. Notice that the minimum squared distance between points within a 2-D subset is 4. Under 90° clockwise rotations A → C → B → D → A. Wei uses the binary labels (Z1, Z0) shown in the following table for these subsets.

SUBSET	Z1	Z0
A	0	0
C	0	1
B	1	0
D	1	1

Table IV-1. 2-D Subset Labels

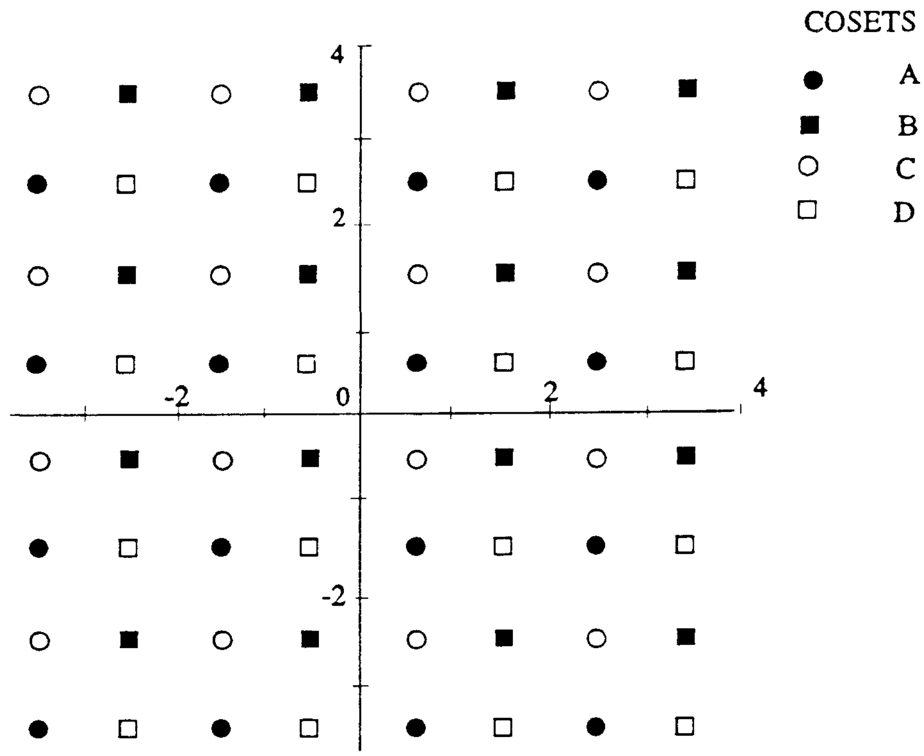

Fig. IV-1. Partitioning of the 2-D Constellation into 4 Cosets

Next, pairs of 2-D subsets are combined to form 4-D subsets. There are 4x4 = 16 possible pairs of 2-D subsets: (A,A), (A,B), ..., (D,D). Wei calls each pair a 4-D type. The 4-D types correspond to cosets of the 4-D lattice $2Z^4 = (A,A) - (\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2})$ in Z^4. He then groups pairs of 4-D types to form the 4-D subsets shown in Table IV-2. The column labels $Y0_n$, $I1_n$, $I2_n'$, and $I3_n'$ in the table refer to signals in the 4-D convolutional encoder shown in Fig. IV-3.

The minimum squared Euclidean distance between points within a 4-D type is 4, the same as for the 2-D subsets. For example, consider the type (A,A). Two minimum distance 4-D points within a type can be formed by taking minimum distance A points for the first baud and the same A point for the second baud or vice versa.

4D Sub-set	Rotation Group $Y0_n$	$I1_n$	Rotation in Group $I2_n'$	$I3_n'$	4D TYPE	$Z0_n$	$Z1_n$	$Z0_{n+1}$	$Z1_{n+1}$
0	0	0	0	0	(A,A)	0	0	0	0
	0	0	0	1	(B,B)	0	1	0	1
1	0	0	1	0	(C,C)	1	0	1	0
	0	0	1	1	(D,D)	1	1	1	1
2	0	1	0	0	(A,B)	0	0	0	1
	0	1	0	1	(B,A)	0	1	0	0
3	0	1	1	0	(C,D)	1	0	1	1
	0	1	1	1	(D,C)	1	1	1	0
4	1	0	0	0	(A,C)	0	0	1	0
	1	0	0	1	(B,D)	0	1	1	1
5	1	0	1	0	(C,B)	1	0	0	1
	1	0	1	1	(D,A)	1	1	0	0
6	1	1	0	0	(A,D)	0	0	1	1
	1	1	0	1	(B,C)	0	1	1	0
7	1	1	1	0	(C,A)	1	0	0	0
	1	1	1	1	(D,B)	1	1	0	1

Table IV-2. Partitioning of the 4-D Rectangular Lattice

The minimum squared Euclidean distance between points in a 4-D subset is also 4. For example, consider subset 0 which consists of the types (A,A) and (B,B). A minimum distance point can be formed by selecting two closest A and B points for the 1st baud and two closest A and B points for the second baud. The squared distance between closest A and B points is 2 so the minimum squared distance between (A,A) and (B,B) is 2+2=4 which is the same as the minimum squared distance in (A,A) and (B,B).

The 4-D points have been partitioned into four **rotation groups** which are specified by the bit pair ($Y0_n$,$I1_n$). 90° rotations leave 4-D points in the same rotation group. For example, in rotation group (0,0), (A,A)→(C,C)→(B,B)→(D,D)→(A,A) under 90° clockwise rotations. The elements of each rotation group are listed in the same rotational order in Table IV-2. The rotation relative to the first element listed in each rotation group is specified by the bit pair ($I3_n'$,$I2_n'$) and is given in Table IV-3. These rotational properties are part of the reason the 4-D code is transparent to 90° rotations.

($I3_n'$,$I2_n'$)	Clockwise Rotation
(0,0)	0°
(0,1)	90°
(1,0)	180°
(1,1)	270°

Table IV-3. Clockwise Rotation Relative to First Element of Rotation Group

The partitioning of the 4-D constellation can be displayed as an Ungerboeck type cf partition tree. This tree is shown in Fig. IV-2. Notice that the minimum squared Euclidean distance within the subsets at the different levels either stays the same or increases from the top to the bottom of the tree. Wei's partitioning is not quite a linear lattice theoretic one. The 4-D types that make up the translated lattice $\Lambda_0 = \mathbf{Z}^4 + (\frac{1}{2},\frac{1}{2})$ can be represented by the formula

$$\Lambda_0 = (A,A) + I3_n'(1111) + I2_n'(0101) + I1_n(0011) + Y0_n(0001) + Y0_n \cdot I2_n'(0011) \qquad \textbf{(IV-1)}$$

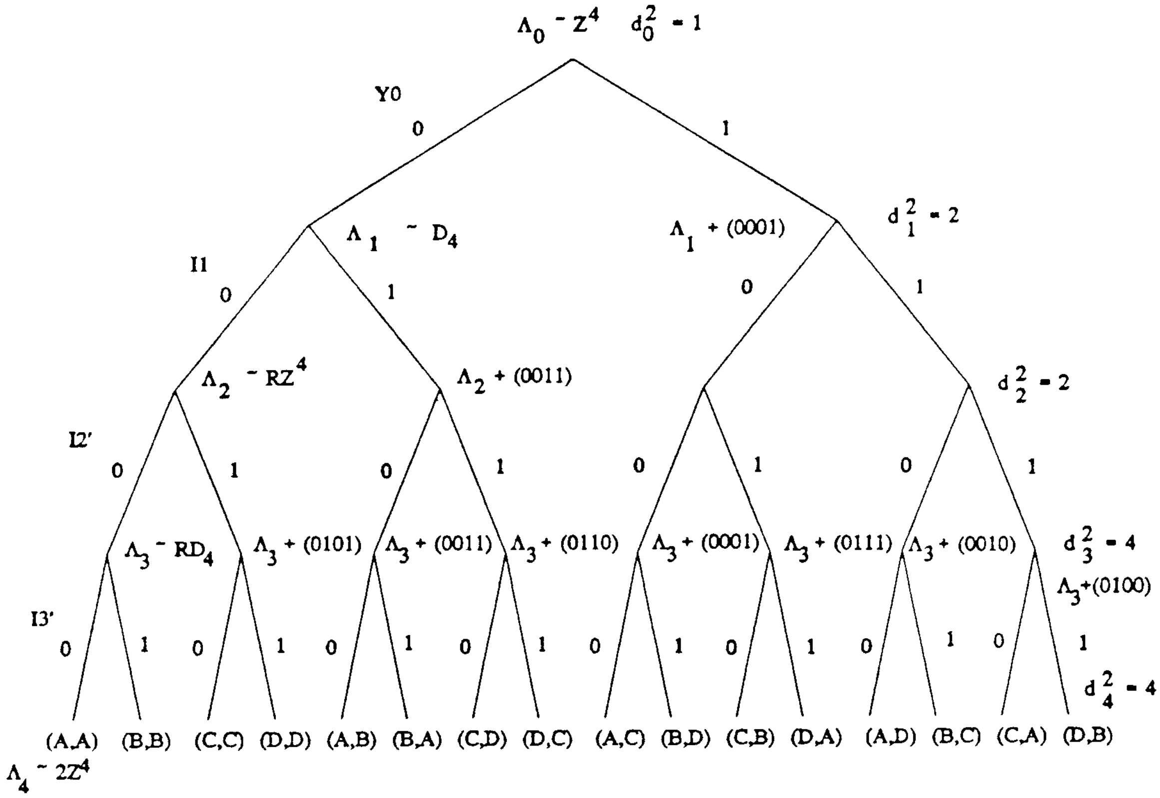

Fig. IV-2. Wei's Partition Tree for the 4-D Lattice

The lattice at level 0 is a translate of the 4-D integer lattice Z^4. The lattice at level 1 is designated by D_4 and is called the Schläfli lattice. It is the densest 4-D lattice. This lattice consists of the set of integer 4-tuples with even squared norm. (The squared norm is the sum of the squares of the components.) Since the square of an odd number is odd, an equivalent definition of D_4 is the set of integer 4-tuples with an even number of odd components, that is, 0, 2, or 4 odd components. The minimum squared Euclidean distance in D_4 is $d_1^2 = 2$.

The subset of D_4 with 0 odd components or, equivalently, all even components is just the lattice $2Z^4$ which is a translate of the 4-D type (A,A) and its coset representative in the partition $Z^4/2Z^4$ is (0000).

There are $\binom{4}{2}=6$ subsets of D_4 with two odd components and one subset with 4 odd components. They are all cosets of $2Z^4$ and have the coset representatives shown in Table IV-4 along with the corresponding 4-D types.

Coset Representative	4-D Type
0000	(A,A)
1100	(B,A)
1010	(D,D)
1001	(D,C)
0110	(C,D)
0101	(C,C)
0011	(A,B)
1111	(B,B)

Table IV-4. Coset Representatives for $D_4/2Z^2$

The coset representatives in Table IV-4 are the 8 even weight binary 4-tuples. The sum of even weight N-tuples is also an even weight N-tuple. Therefore, the union of the 8 cosets of $2Z^4$ formed from these representatives must form a lattice which we have designated by D_4. The remaining eight cosets of $2Z^4$ in Z^4 formed from the representatives which are the 8 odd weight 4-tuples correspond to the 8 4-D types not listed in Table IV-4 and their union is the coset D_4 + (0001) of D_4.

The number of nearest neighbors to any point in D_4 is 24. This is because nearest neigbors to a point are formed by adding the 6 coset representatives with two 1's in Table IV-4 and their sign permutations to the point.

Since $D_4 \cup [D_4 + (0001)]$ includes the 16 cosets of $2Z^4$ in Z^4, Z^4/D_4 is a lattice partition of order 2. Thus the fundamental volume of D_4 is $V(D_4) = 2\ V(Z^4) = 2$. The points in the lattice $RZ^4 = [RZ^2, RZ^2]$ all have even squared norm, so RZ^4 is a sublattice of D_4 and we have the lattice partition chain $Z^4/D_4/RZ^4$. Since $|Z^4/RZ^4| = 4$ and $|Z^4/D_4| = 2$, it follows that $|D_4/RZ^4| = 2$. Applying the R operator to the three lattice chain two lines above gives the further partition chain

$Z^4/D_4/RZ^4/RD_4/2Z^4$

with minimum squared Euclidean distances

1/2/2/4/4

which is exactly the chain shown in Fig. IV-2.

Remember that the fundamental coding gain for an N-dimensional lattice was defined as

$$\gamma(\Lambda) = \frac{d^2}{V^{2/N}(\Lambda)} \qquad \textbf{(IV-2)}$$

Therefore

$$\gamma(D_4) = \frac{2}{2^{2/4}} = \sqrt{2} = 1.52\ dB \qquad \textbf{(IV-3)}$$

B. The 4-D Trellis Encoder

A block diagram for Wei's 16-state 4-D trellis encoder is shown in Fig. IV-3. Every 2 bauds, the encoder accepts $L = 3 + n_u$ data bits where n_u is the number of uncoded data bits. Two of the coded bits $I2_n$ and $I3_n$ are differentially encoded resulting in the pair $I2_n'$ and $I3_n'$ according to the rule

$$(I3_n', I2_n') = (I3_{n-2}', I2_{n-2}') + (I3_n, I2_n)\ mod\ 4 \qquad \textbf{(IV-4)}$$

where the bit pairs in parentheses are considered to be 2-bit positive binary numbers with the most significant bit on the left. The decoder contains a complementary differential decoder with the

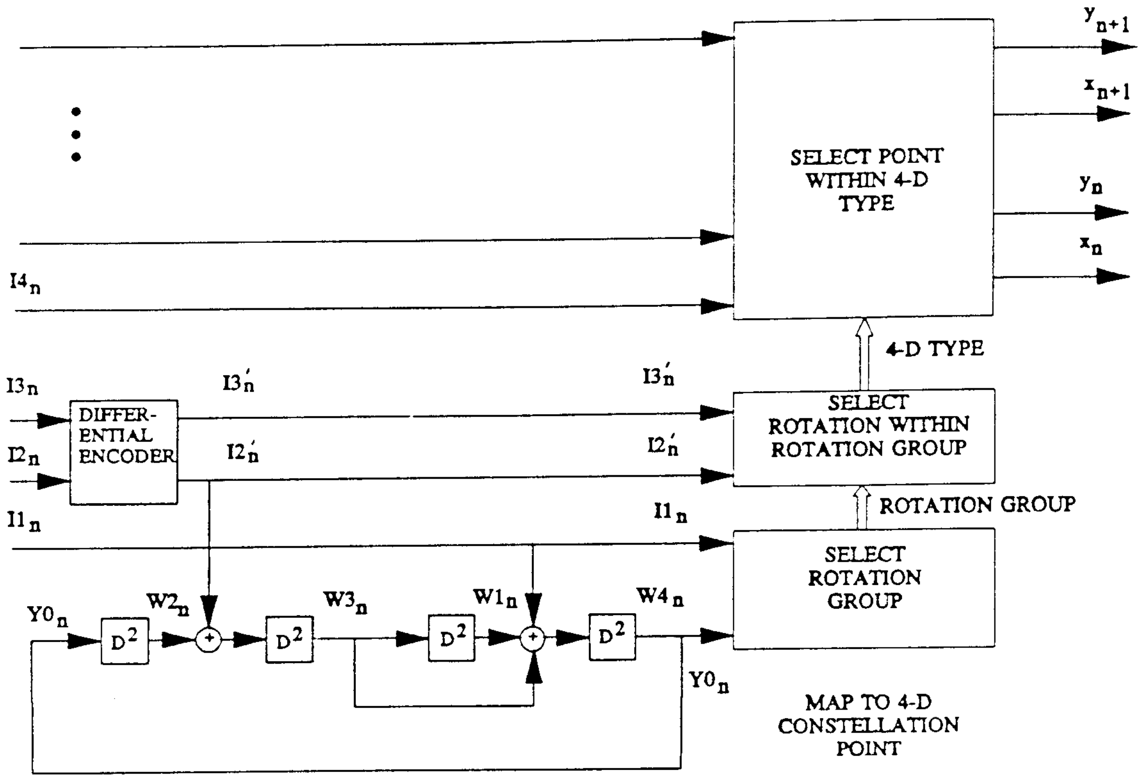

Fig. IV-3. The Encoder for Wei's 4-D, 16-State Code

rule

$$(I3_n, I2_n) = (I3_n', I2_n') - (I3_{n-2}', I2_{n-2}') \ mod\ 4 \qquad \textbf{(IV-5)}$$

The two bits $I1_n$ and $I2_n'$ enter the (3,2) convolutional encoder at the bottom of the figure resulting in the parity check symbol $Y0_n$. The D^2 blocks represent 2-baud delays. The bit pair $(Y0_n, I1_n)$ selects a rotation group as specified in Table IV-2 and $(I2_n', I3_n')$ selects a rotation within the group. The result of these two operations is the selection of a 4-D type. These two operations can be simultaneously performed by a simple table look-up based on Table IV-2. The remaining uncoded bits select a specific 4-D point within the 4-D type. This selection can be performed in a variety of ways and has little effect on the coding gain.

The trellis for this convolutional code is described by Table IV-5. The entries in the table are $(W1_{n+2}, W2_{n+2}, W3_{n+2}, W4_{n+2})$ which is the next encoder state. Notice that $Y0_n = W4_n$ is one encoder output and the other two outputs are the inputs $I1_n$ and $I2_n'$. Also, the 4-D subset selected by the encoder is $(Y0_n, I1_n, I2_n')$.

Current State $W1_n, W2_n, W3_n, W4_n$	Input $(I1_n, I2_n')$			
	00	01	10	11
0000	0000	0010	0001	0011
0001	0100	0110	0101	0111
0010	1001	1011	1000	1010
0011	1101	1111	1100	1110
0100	0010	0000	0011	0001
0101	0110	0100	0111	0101
0110	1011	1001	1010	1000
0111	1111	1101	1110	1100
1000	0001	0011	0000	0010
1001	0101	0111	0100	0110
1010	1000	1010	1001	1011
1011	1100	1110	1101	1111
1100	0011	0001	0010	0000
1101	0111	0101	0110	0100
1110	1010	1000	1011	1001
1111	1110	1100	1111	1101

Table IV-5. State Transitions for the 16-State 4-D Wei Code

To update the cumulative path metrics in the Viterbi decoding algorithm, knowledge of which four previous states have branches that connect with a current state is required. This is shown in the following table. The entries in the table are the previous state ($W1_{n-2}, W2_{n-2}, W3_{n-2}, W4_{n-2}$) and 4-D subset ($Y0_{n-2}, I1_{n-2}, I2'_{n-2}$) for the transition from the previous to current state.

Current State $W1_2, W2_n, W3_n, W4_n$	Previous States and Subsets			
0000	0000,000	0100,001	1000,010	1100,011
0001	0000,010	0100,011	1000,000	1100,001
0010	0000,001	0100,000	1000,011	1100,010
0011	0000,011	0100,010	1000,001	1100,000
0100	0001,100	0101,101	1001,110	1101,111
0101	0001,110	0101,111	1001,100	1101,101
0110	0001,101	0101,100	1001,111	1101,110
0111	0001,111	0101,110	1001,101	1101,100
1000	0010,010	0110,011	1010,000	1110,001
1001	0010,000	0110,001	1010,010	1110,011
1010	0010,011	0110,010	1010,001	1110,000
1011	0010,001	0110,000	1010,011	1110,010
1100	0011,110	0111,111	1011,100	1111,101
1101	0011,100	0111,101	1011,110	1111,111
1110	0011,111	0111,110	1011,110	1111,100
1111	0011,101	0111,100	1011,111	1111,110

Table IV-6. Previous States that Reach a Current State

Since the two encoder inputs are not connected to adders at the output of the right-hand D^2 element or input of the left-hand D^2 element, the four transitions from a given state all have the same $Y0_n$ and thus come from the same subset at level 1. Also, the branches leading into a state have the same $Y0_n$. Using this property, it can be shown that the minimum squared Euclidean distance along divergent paths in the trellis is greater than the 4-D subset minimum squared Euclidean distance. Therefore, the minimum free distance for this code is $d_f^2 = d_4^2 = 4$. In "Coset Codes I," on page 1140 Forney defines the coding gain of an N-dimensional trellis code based on a lattice partition Λ/Λ' and (n,k) convolutional code C with redundancy r = n-k to be

$$\gamma = \frac{d_f^2}{2^{2r/N} V^{2/N}(\Lambda)} \qquad \textbf{(IV-6)}$$

Thus, for this code which has r=1, $\Lambda = \mathbf{Z}^4$, and $V(\mathbf{Z}^4) = 1$

$$\gamma = \frac{4}{2^{2/4} \times 1^{2/4}} = 2\sqrt{2} = 4.52\ db \qquad \textbf{(IV-7)}$$

This code is transparent to 90° rotations. Under 90° rotations, transmitted points remain in the same rotation group so $(Y0_n, I1_n)$ is unaffected. Under a k*90° clockwise rotation, $(I3_n', I2_n')$ considered as a 2-bit positive binary number changes to $(I3_n', I2_n') + k \bmod 4$. The differential encoder in the transmitter and differential decoder in the receiver make the output depend only on the phase change between adjacent 4-D symbols which is unaffected by k. Thus, without channel errors, the decoded output values for $(I3_n, I2_n)$ will be transparent to the rotations.

Finally, we must check that the rotated sequences are consistent with the convolutional encoder trellis. From the encoder block diagram it can be seen that codewords satisfy the parity check equation

$$Y0_{n+2} = W3_n + W1_n + I1_n = Y0_{n-4} + I2'_{n-2} + Y0_{n-6} + I2'_{n-4} + I1_n \qquad \textbf{(IV-8)}$$

Under a 90° clockwise rotation,

$$(I3'_n, I2'_n) \rightarrow (I3'_n, I2'_n) + (0,1) \ mod\ 4 = (I3'_n \oplus I2'_n,\ I2'_n \oplus 1) \qquad \textbf{(IV-9)}$$

Also $(Y0_n, I1_n)$ is unchanged. Substituting these new values in the parity check equation, (IV-8), we see that it is still satisfied, so the sequence is consistent with the trellis.

C. The 4-D Trellis Decoder

Viterbi decoding of Wei's 16-state 4-D code is almost the same as Viterbi decoding of 2-D codes. The required decoding steps are:

1. **Quantizing received 4-D points to 4-D subsets**

   The first step is to quantize the received 4-D point consisting of a pair of consecutive 2-D received points to the nearest ideal point in each of the eight 4-D subsets listed in Table IV-2 on page 61. This can be done with a 2-stage process.

   Stage 1. Quantizing to 2-D subsets

   First quantize the first 2-D received symbol to the nearest A, B, C, and D subset 2-D points and record (1) the squared Euclidean errors between the received and quantized points, (2) $Z0_n, Z1_n$, and (3) the corresponding uncoded bits. Repeat this process for the 2nd received 2-D point.

   Stage 2. Quantizing to 4-D subsets

   Next quantized 2-D subsets are combined to form quantized 4-D subsets. For each 4-D subset listed in Table IV-2 compute the squared Euclidean error for each of the two 4-D types by adding the squared errors saved in Stage 1 for the 1st and 2nd received 2-D points. Select the type from the pair with the smallest 4-D error and save the 4-D squared branch error and Z bits. For example, for 4-D subset 0, compute the squared 4-D errors for 4-D types (A,A) and (B,B) and select the one with the smallest squared error.

This process can occasionally lead to 4-D points that do not lie in the finite point constellation. This typically happens rarely and most likely with points on the boundary of the constellation. The easiest way to handle this event is to simply let the decoder make an error. More sophisticated schemes can be easily invented. The required procedure depends on how the uncoded input bits were assigned to constellation points.

2. **Updating the cumulative path metrics**
   The standard method for updating the cumulative path metrics can be used. Each of the eight trellis states has four branches that lead to it. These are listed in Table IV-6 on page 68 along with the 4-D subsets associated with the branches. The decoder must contain a list of the eight cumulative path metrics for the survivors to each state. For a given state, the four branch metrics (squared Euclidean errors) for the paths converging on that state should be added to the cumulative path metrics for the states the branches come from. The surviving path to the given state is the one corresponding to the smallest of the four cumulative path metrics to the given state. The surviving cumulative path metric should be stored in the updated path metric table and a pointer to the best previous state should be stored in the trellis record for the given state along with the Z bits and uncoded bits.
3. **Tracing back the trellis path**
   The decoder output is found by first selecting the current state with the smallest updated surviving cumulative path metric. Then the path starting at this state can be traced back to the end of the trellis memory by following the pointers to the best previous states. The four Z bits found at the record at the end of the trellis can be converted back to the encoder input bits Y0, I1, I2′, and I3′ using a 16 word lookup table based on Table IV-2 and (I3′,I2′) can be differentially decoded according to Equation IV-5.

D. Trellis Shaping Combined with Wei's 4-D Code

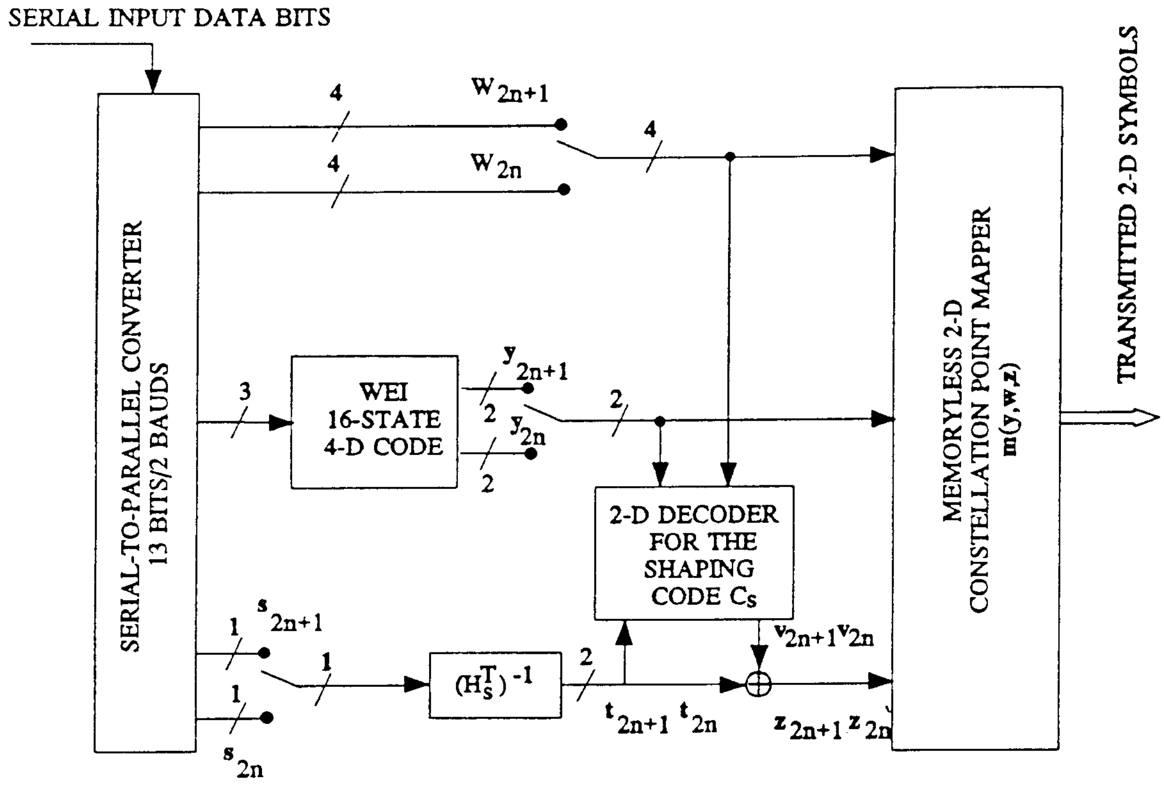

Fig. IV-4. Combining Trellis Shaping with Wei's 4-D Code

The trellis shaping methods presented in Sections III-B and III-C can be combined with Wei's 4-D code quite easily. The technique for doing this will be described by using the example shown in Fig. IV-4 above. Each 4-D symbol (every 2 bauds), the combined encoder takes in 13 source data bits which are stored in the Serial-to-Parallel Converter, so the data rate is 6.5 bits/baud. Three of these bits are selected for the 4-D Wei encoder inputs $I1_n$, $I2_n$, and $I3_n$ shown in Fig. IV-3. The encoder selects a 4-D type consisting of the pair of 2-D subsets $\mathbf{y}_{2n}$ = $(Z1_{2n}, Z0_{2n})$ and $\mathbf{y}_{2n+1} = (Z1_{2n+1}, Z0_{2n+1})$ shown at the output of the Wei encoder in Fig. IV-4. The subscript, n, is the time index in 2-D symbols. Eight of the input bits are uncoded and are grouped into the two 4-tuples $\mathbf{w}_{2n}$ and $\mathbf{w}_{2n+1}$. Without shaping, the pair of 6-tuples, $(\mathbf{y}_{2n}, \mathbf{w}_{2n})$ and $(\mathbf{y}_{2n+1}, \mathbf{w}_{2n+1})$, specify a pair of 2-D points, each selected from a 64-point constellation consisting of 16 points specified by the $\mathbf{w}$'s from each of the four 2-D subsets specified by the $\mathbf{y}$'s. For example, this constellation could be the 64 points on the 8x8 grid in the 1st quadrant of the $\mathbf{Z}^2$ lattice shown in Fig. IV-1 with the $\mathbf{w}$ assignment shown in Fig. III-12. The remaining two bits labelled s_{2n} and s_{2n+1} are connected to the inverse syndrome former for the trellis shaping code.

The shaping is performed on 2-D symbols using the Ungerboeck 4-state code as described in Example III-10. Remember that the 2-D shaping code was based on the lattice partition $\Lambda_s/\Lambda_s' = 8Z^2/16Z^2$ and the 4-D code selects pairs of 2-D symbols from shifted cosets of $2Z^2$. Thus, shaping can be performed serially on the stream of 2-D symbols selected by the 4-D channel encoder since $8Z^2$ is a sublattice of $2Z^2$ so A, B, C, and D are invariant to translations by elements of $8Z^2$. Each baud (2-D symbol) the output $\mathbf{t}$ of the inverse syndrome former, one of the outputs $\mathbf{y}$ of the Wei 4-D encoder, and one of the pair of uncoded 4-tuples $\mathbf{w}$ enters the Decoder for C_s which selects the best shaping code sequence as described in Section III-C. The resulting 8-tuple $(\mathbf{z}_{2n}, \mathbf{y}_{2n}, \mathbf{w}_{2n})$ or $(\mathbf{z}_{2n+1}, \mathbf{y}_{2n+1}, \mathbf{w}_{2n+1})$ is mapped to a 2-D point selected from a 256-point constellation. The code can be made 90° rotationally invariant using the constellation described in Example III-10.