
Two-Dimensional Audio Watermark for MPEG AAC Audio

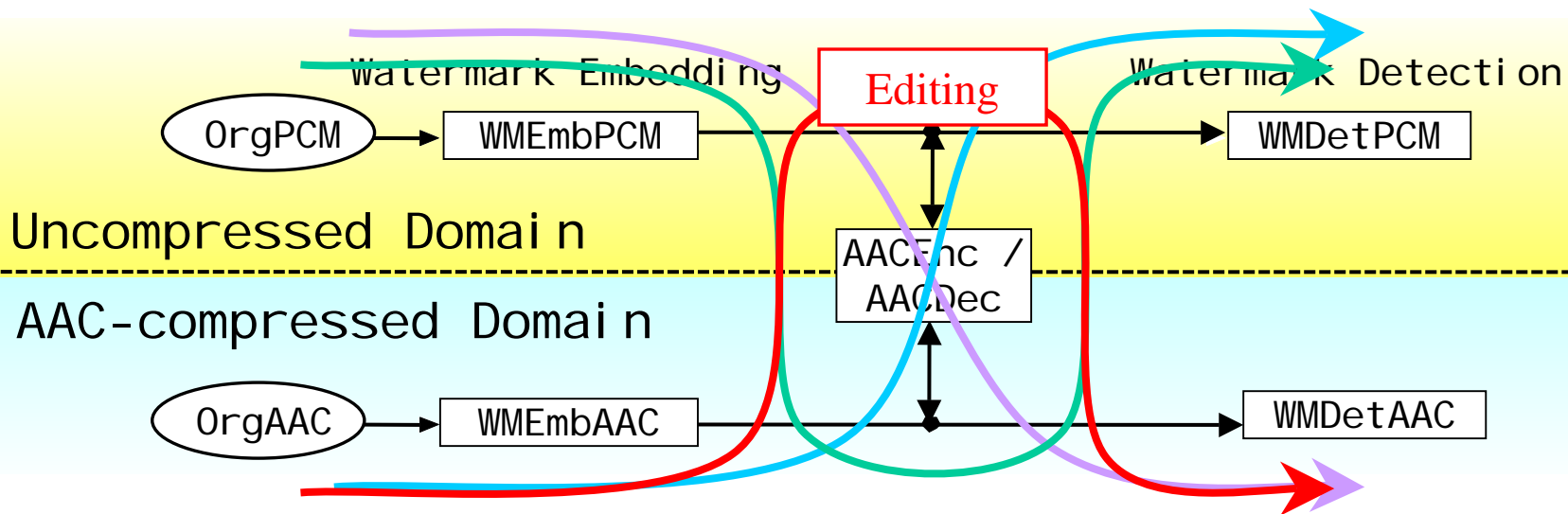
Ryuki Tachibana

Tokyo Research Laboratory

IBM Japan

Purpose

Compatibility between the uncompressed-domain watermarking and the AAC-compressed-domain watermarking

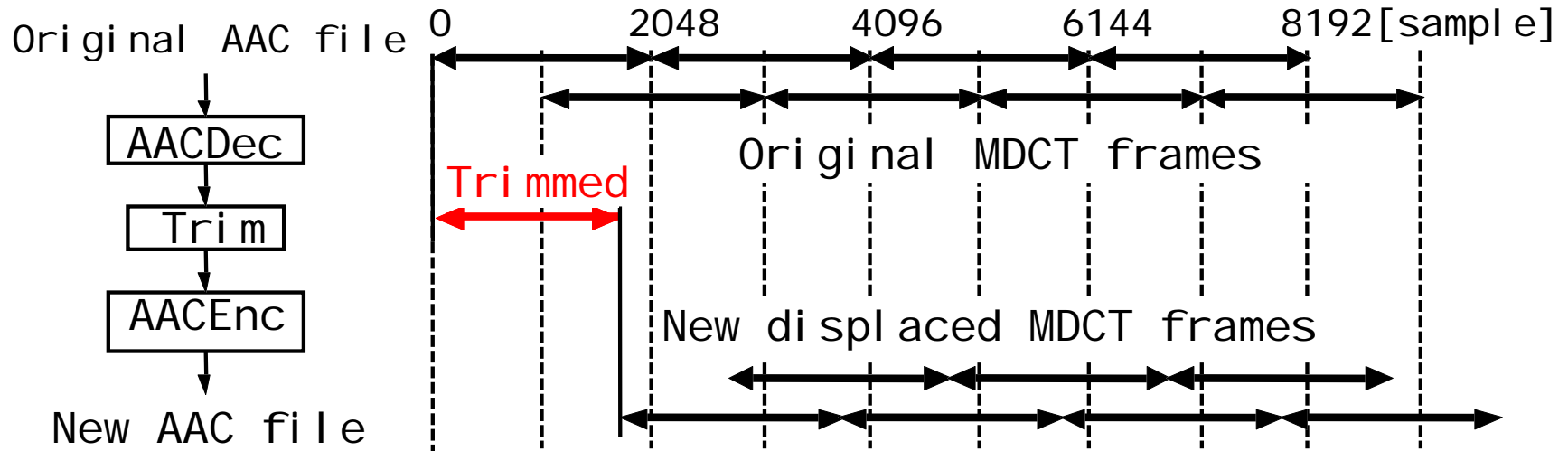


Other aspects of our research project:

- (1) Robust and transparent watermark
- (2) Blind detection
- (3) Multiple-bit embedding
- (4) Frequency-domain Spread Spectrum

Problem

The detection MDCT frames are not necessarily identical to the original MDCT frames.



Basic Concepts of the Method

- Use Magnitudes for Watermarking
- Use Two-Dimensional Pseudo-Random Array

① Use Magnitudes for Watermarking

Magnitude changes can be observed in both the uncompressed domain and the AAC-compressed domain.

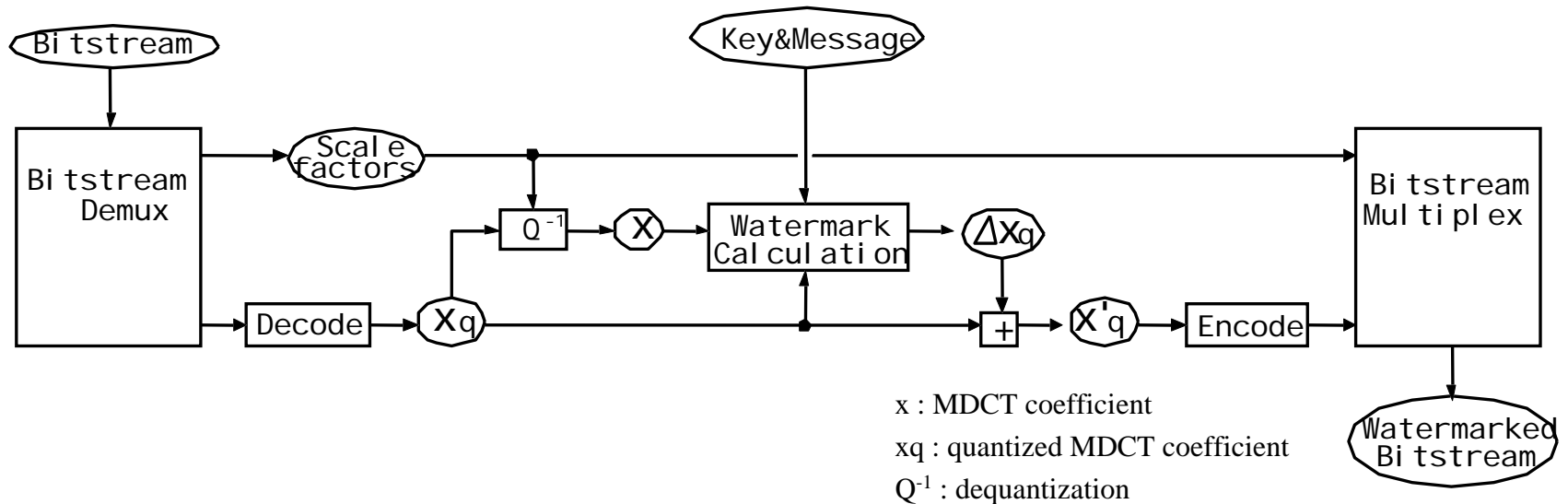
- Embedding introduces statistical non-uniformity to the magnitude distribution of the content.
- Detection observes the non-uniformity using correlation calculation.

“Magnitudes” are defined as:

- Amplitudes of the complex DFT coefficients for the uncompressed domain
- Absolute values of the MDCT coefficients for the AAC-compressed domain

AAC Watermark Embedding

The quantized MDCT coefficients are modified.

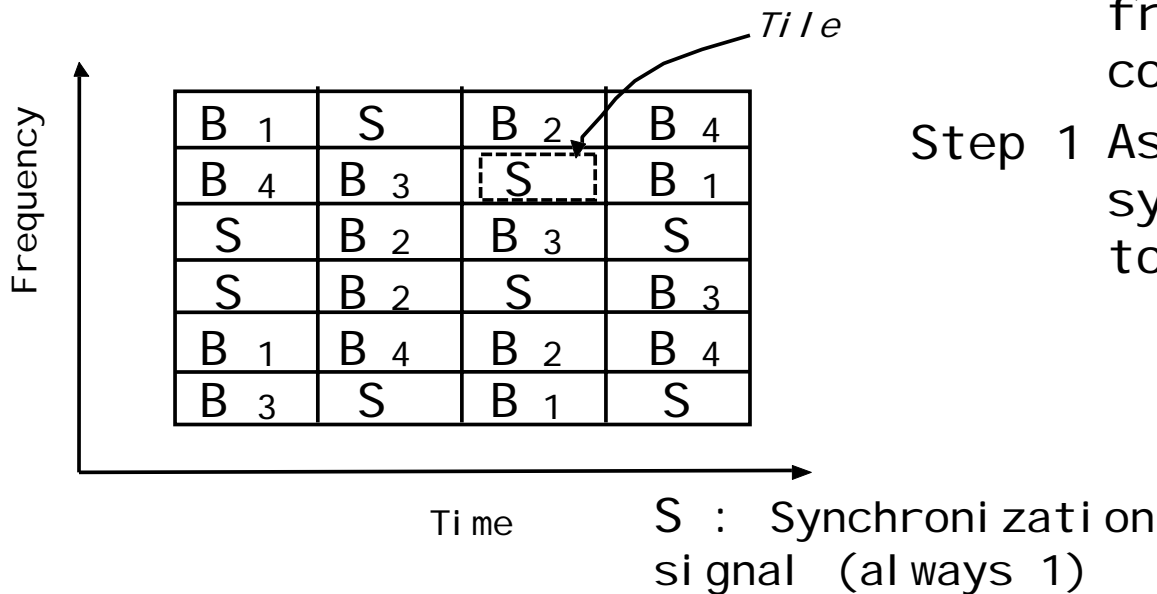


- Re-quantization unnecessary
- Dequantization required to calculate watermark signal
 - Decode the quantized MDCT coefficients
 - Dequantize the MDCT coefficients
 - Determine how many coefficients are to be modified
 - Change the quantized coefficients

② Two-Dimensional Pseudo Random Array

Step 0 Divide the time-frequency domain of the content to *tiles*

Step 1 Assign a bit or a synchronization signal to each tile



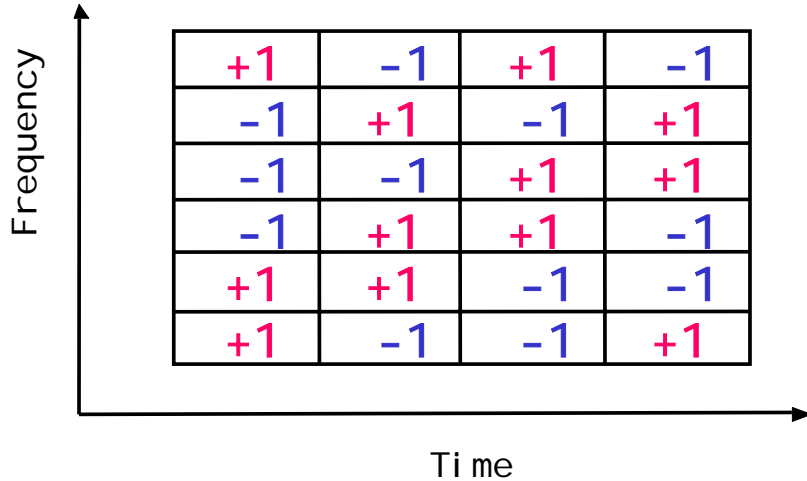
- The magnitudes are modified according to the array.
 - Increased in the tiles with a positive sign.
 - Decreased in the tiles with a negative sign.
- The array has a regularity in its list of the numbers.
 - The opposite signs are used for the first two frames and the last two frames in each tile

② Two-Dimensional Pseudo Random Array

Step 0 Divide the time-frequency domain of the content to *tiles*

Step 1 Assign a bit or a synchronization signal to each tile

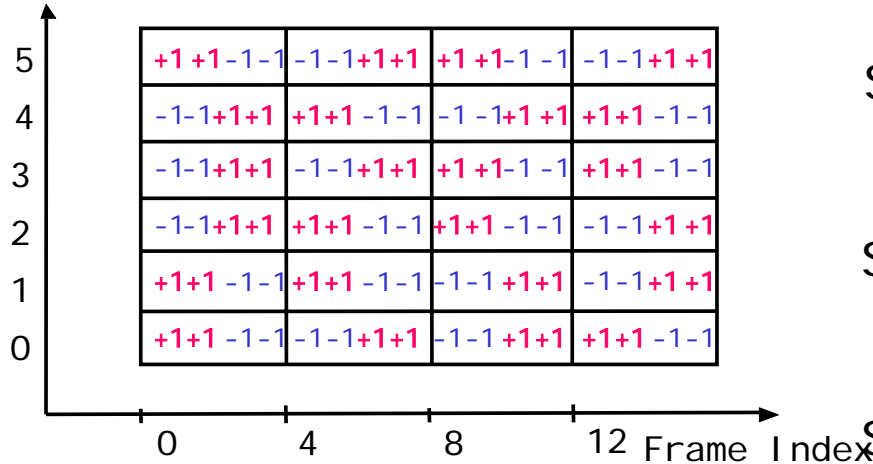
Step 2 Multiply the two-dimensional pseudo-random array



- The magnitudes are modified according to the array.
 - Increased in the tiles with a positive sign.
 - Decreased in the tiles with a negative sign.
- The array has a regularity in its list of the numbers.
 - The opposite signs are used for the first two frames and the last two frames in each tile

② Two-Dimensional Pseudo Random Array

Subband Index (b)



Step 0 Divide the time-frequency domain of the content to *tiles*

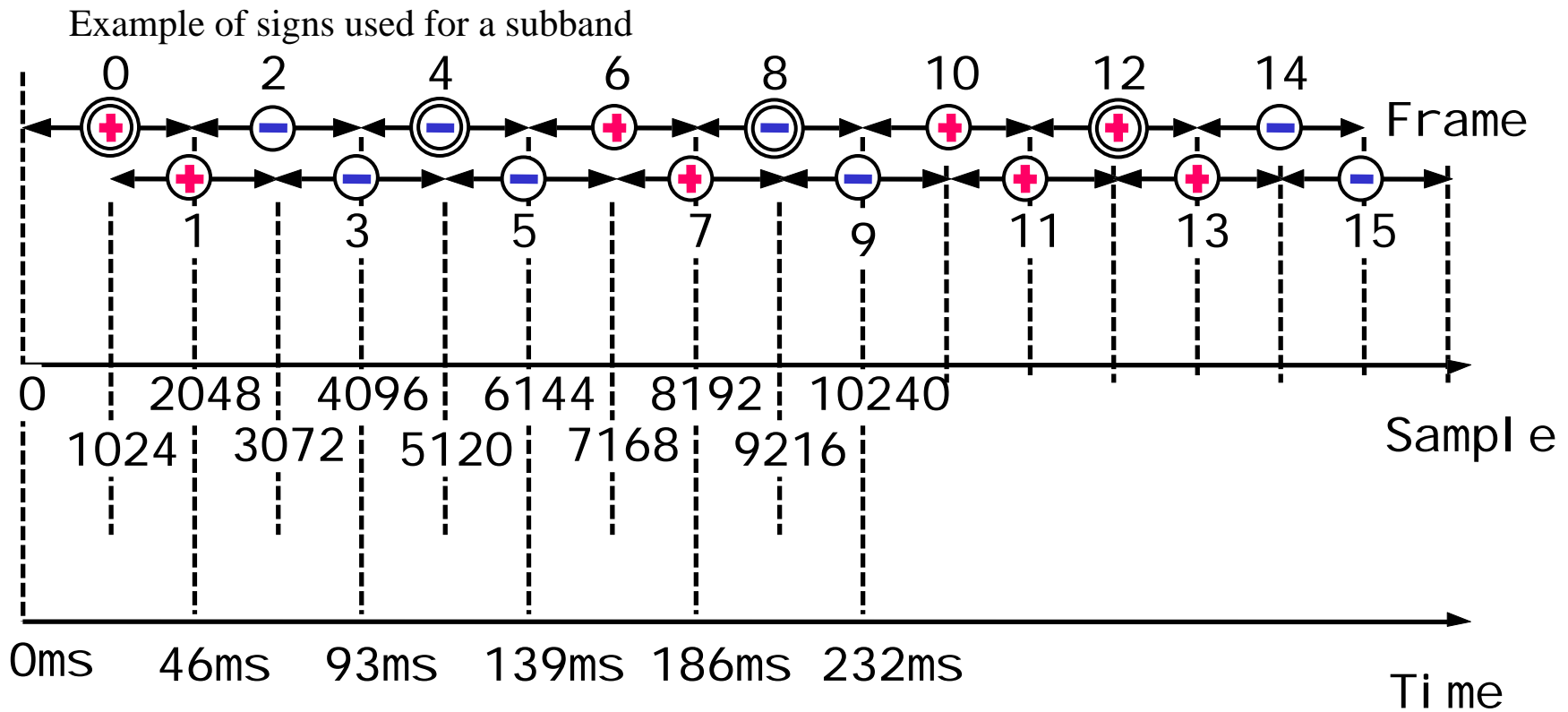
Step 1 Assign a bit or a synchronization signal to each tile

Step 2 Multiply the two-dimensional pseudo-random array

Step 3 Calculate signs that determines magnitude modification in each frame

- The magnitudes are modified according to the array.
 - Increased in the tiles with a positive sign.
 - Decreased in the tiles with a negative sign.
- The array has a regularity in its list of the numbers.
 - The opposite signs are used for the first two frames and the last two frames in each tile

Frames and Signs for Embedding



- The frames overlap with the adjacent frames by a half window.

Detection

- Calculates the correlation between the pseudo-random numbers and the difference of the logarithmic magnitudes
- Synchronize at the strongest frame

$$X = \frac{\sum_{\text{assigned tile}} \varpi_{t,b} (\log P_{t,b} - \log P_{t+2,b})}{\sqrt{\sum_{\text{assigned tile}} \{\varpi_{t,b} (\log P_{t,b} - \log P_{t+2,b})\}^2}}$$

X : Detected Watermark Strength

$\varpi_{t,b}$: Pseudo-Random Number

$P_{t,b}$: The average magnitude in the first frame of a tile

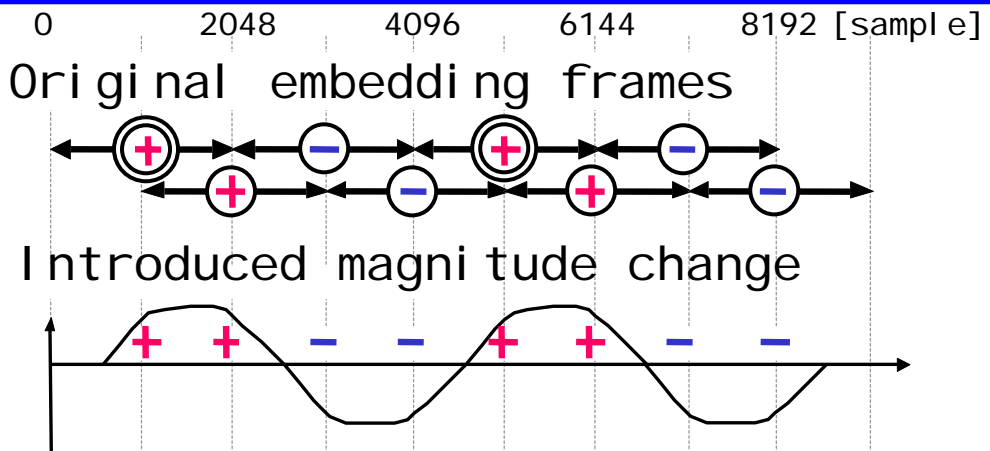
$P_{t+2,b}$: The average magnitude in the third frame of a tile

Detection at the displaced MDCT frames

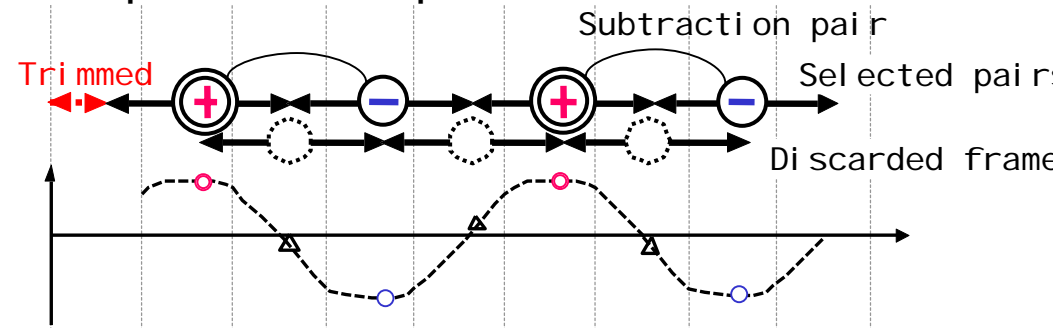
The synchronization process

- selects a set of subtraction pairs (that gives the maximum synchronization strength)
- discards the other set of pairs.

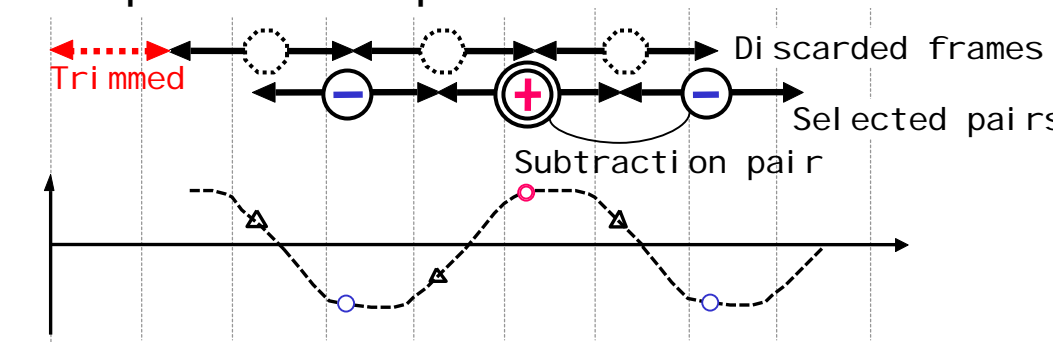
The magnitude changes can be observed strongly from either set of pairs.



Example 1 : Displaced detection frame



Example 2 : Displaced detection frame



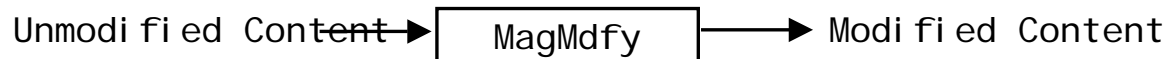
Preliminary Experiment

Magnitude Modification and Observation

Is it possible to observe magnitude changes made in the different domain?

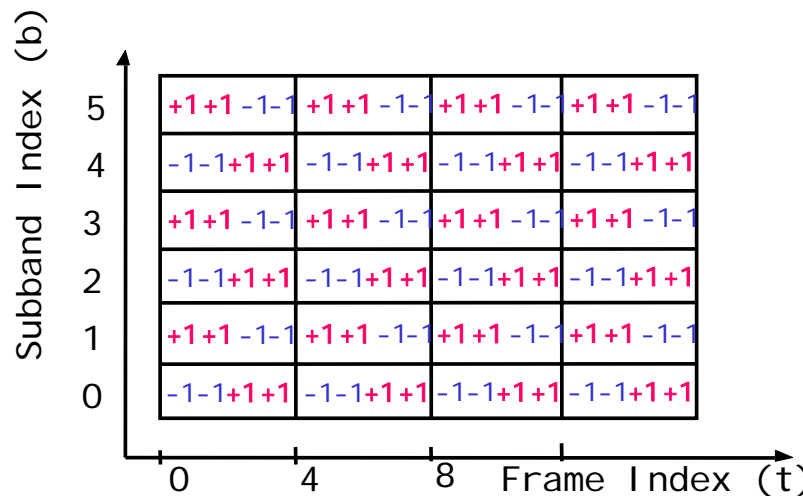
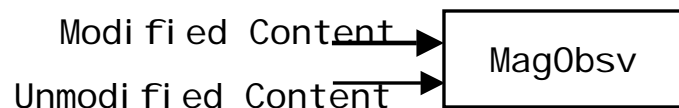
- Magnitude Modification

- The magnitudes of the coefficients are modified by $\pm 10\%$ in either domain.
- A simple checkerwise pattern is used.

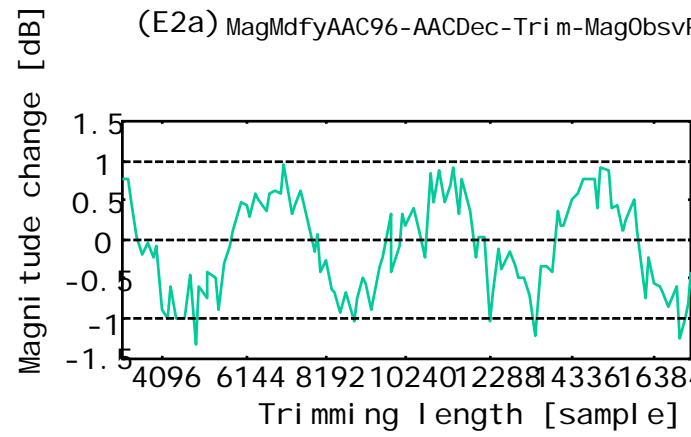
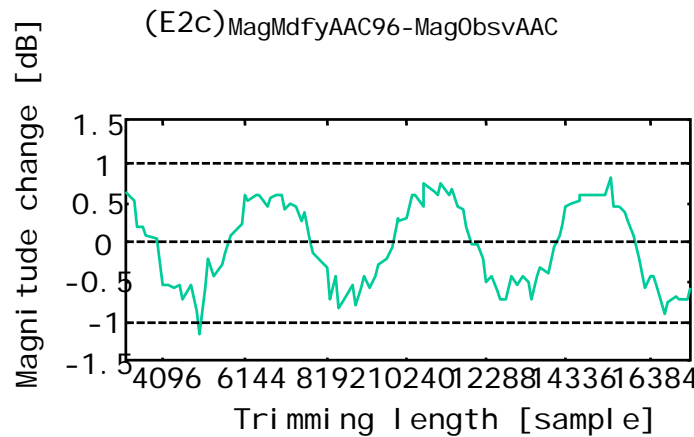
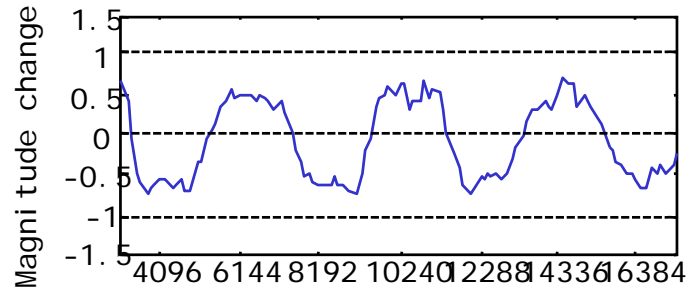
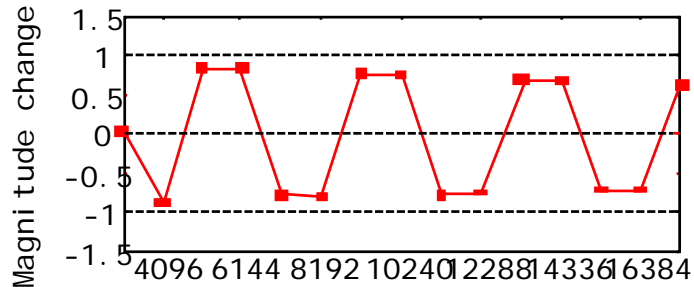
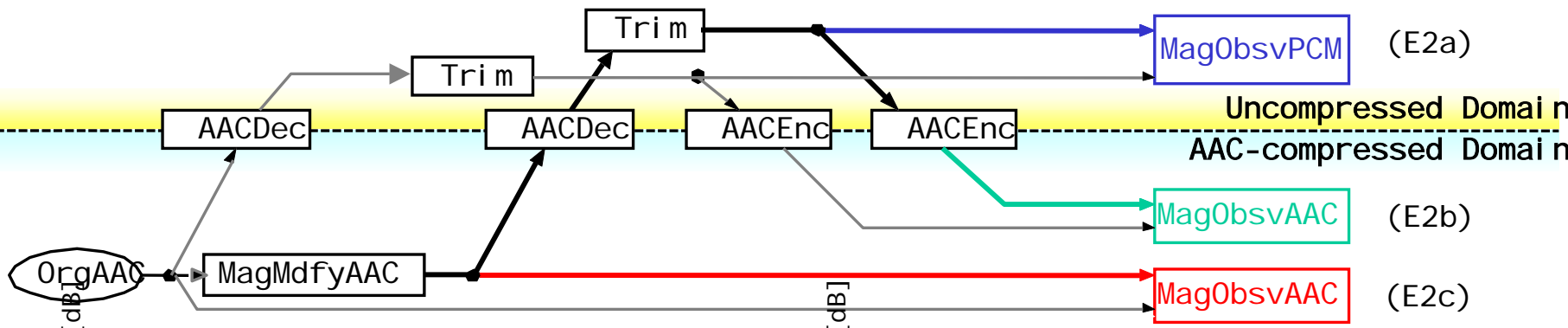


- Magnitude Observation

- The transform windows are displaced by trimming the beginning part of the content before the observation.
- The logarithmic magnitudes with and without magnitude modification are compared.



Magnitude Modification in the AAC-Compressed Domain



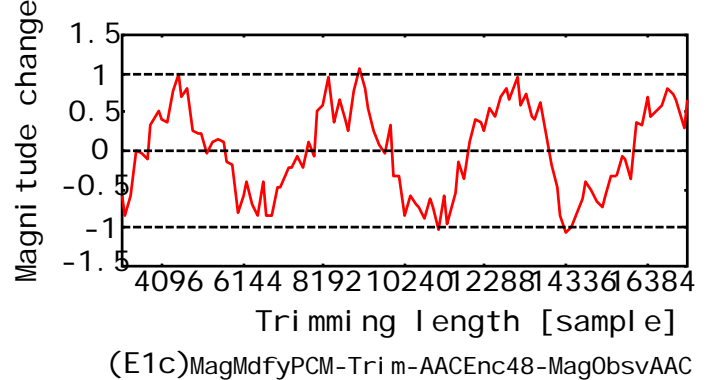
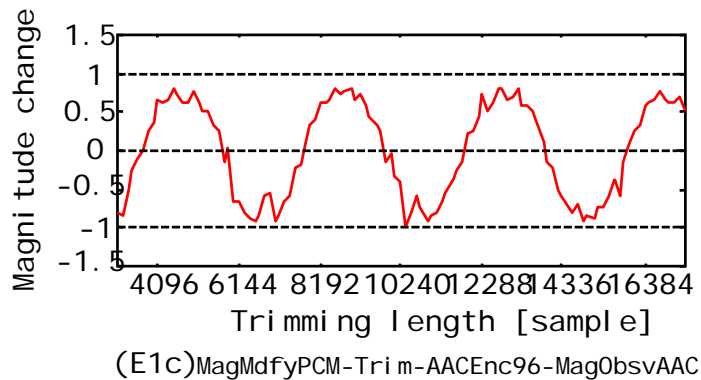
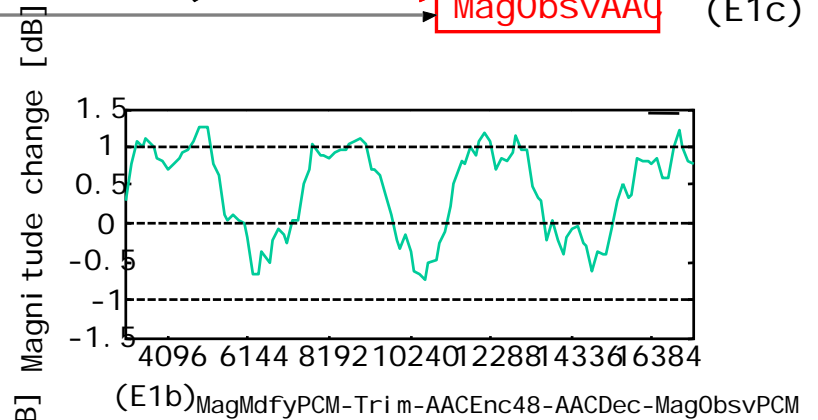
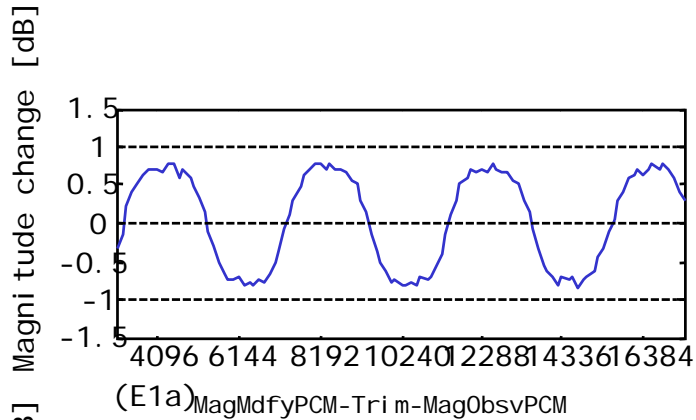
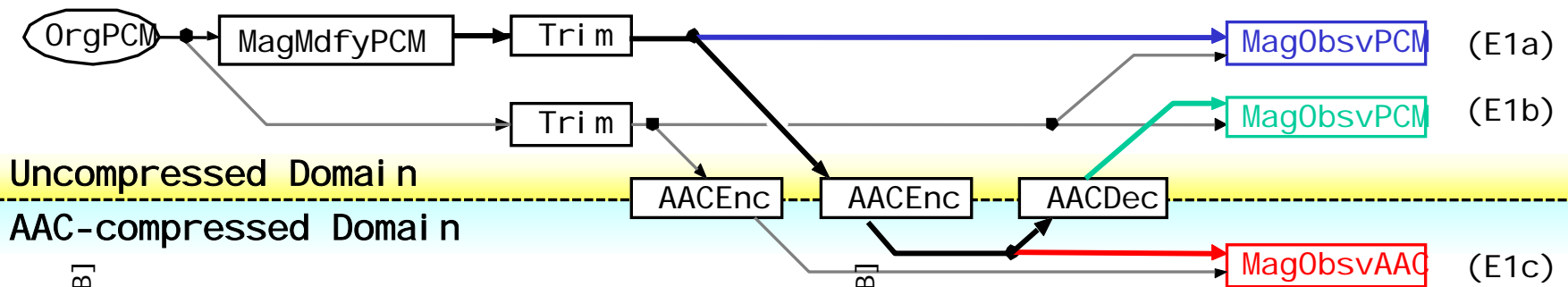
(E2c) **MagMdfyAAC96-MagObsvAAC**

(E2a) **MagMdfyAAC96-AACDec-Trim-MagObsvPCM**

(E2b) **MagMdfyAAC96-AACDec-Trim-AACEnc96-MagObsvAAC**

(E2b) **MagMdfyAAC96-AACDec-Trim-AACEnc48-MagObsvAAC**

Magnitude Modification in the Uncompressed Domain



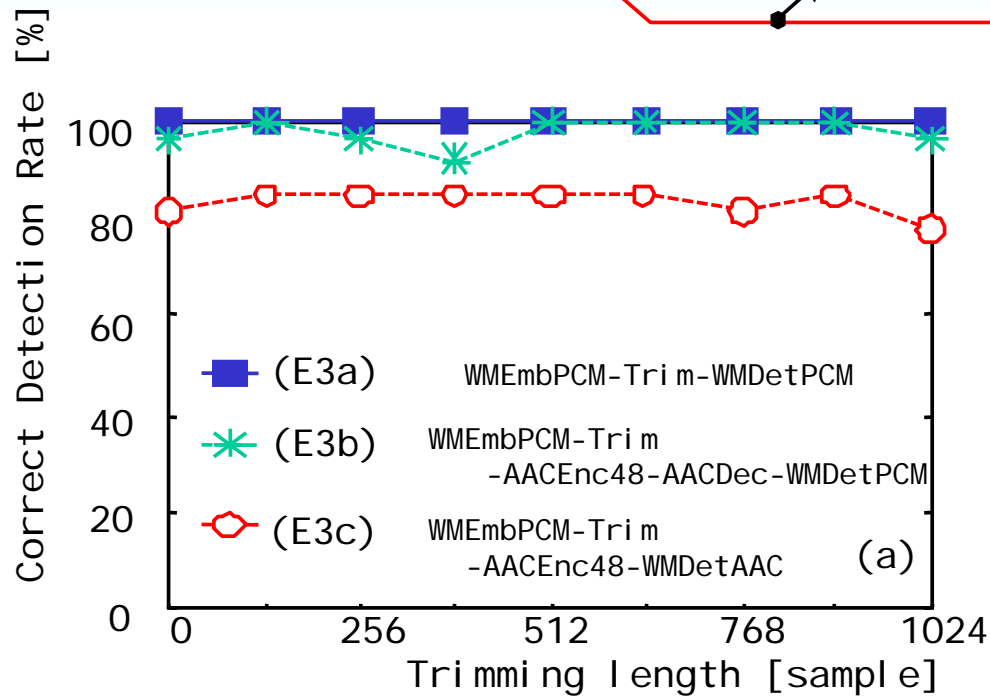
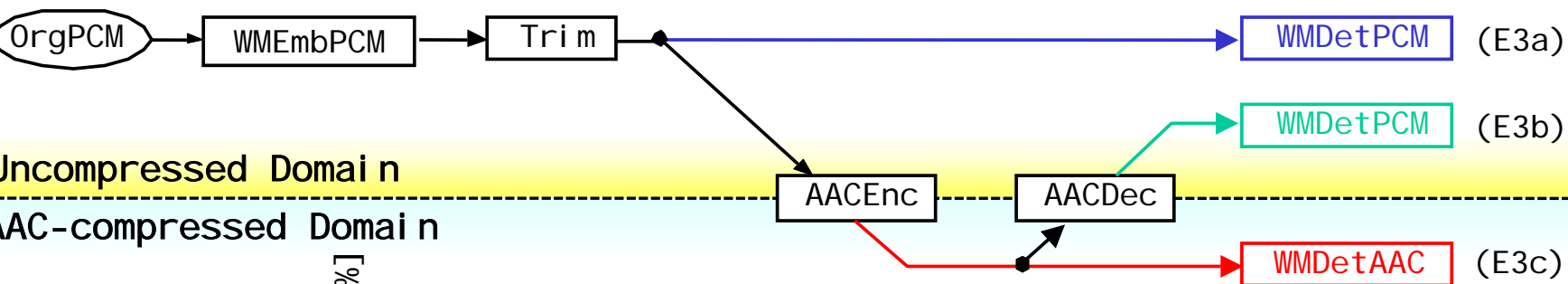
Robustness Test

Is it possible to detect watermark in the different domain?

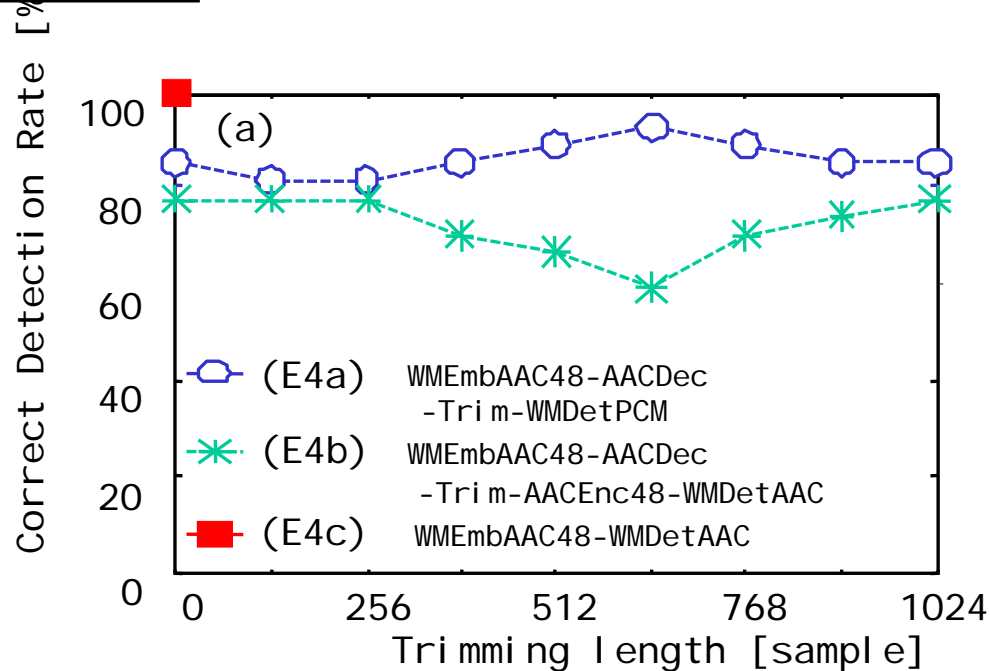
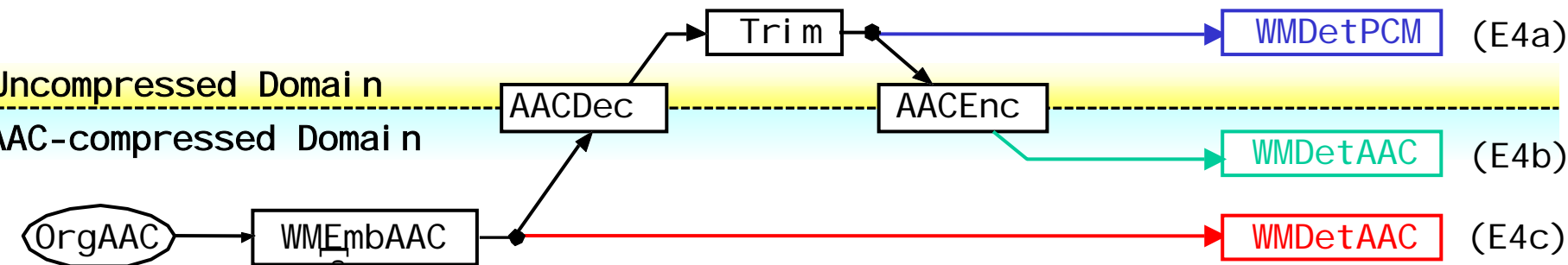
- 64-bit message embedded in 30-second music pieces
- Message encoded in 448 signal bits
 - Cyclic Redundancy Check (CRC) parity bits
 - Turbo Coding
 - Repetition
- The correct detection rate (CDR) of the message observed.

	48 kbps	64 kbps	96 kbps	--
SNR of PCM watermark	--	--	--	23.9 dB
SNR of AAC watermark	19.3 dB	22.0 dB	26.6 dB	--
SNR of AAC encoding	18.1 dB	23.4 dB	29.9 dB	--

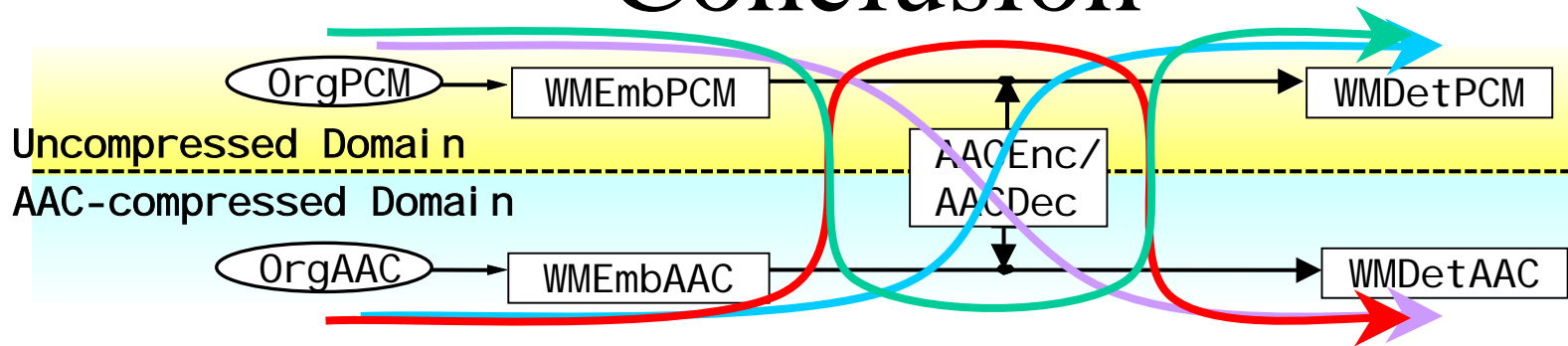
Robustness of Watermark Embedded in the Uncompressed Domain



Robustness of Watermark Embedded in the AAC-Compressed Domain



Conclusion



- Goal
 - Watermark detection regardless of the original embedding domain
- Method
 - Modifying and observing the absolute values of the recovered MDCT coefficients
 - Two-dimensional pseudo-random array with regularity in its list of the numbers
- Experimental Results
 - 80 – 100% correct detection for PCM embedding and AAC detection
 - 50 – 100% correct detection for AAC detection after re-compression
- Future work
 - Acoustic quality
 - Robustness against other audio processing
 - Performance