재귀적 SPCPC에 반복적 복호법을 적용할 때 처리 이득이 성능에 미치는 영향

준회원 전 수 원*, 정회원 김 용 철*

Effect of Processing Gain on the Iterative Decoding for a Recursive Single Parity Check Product Code

Su Won Chon* *Associate Member*, Yong Cheol Kim* *Regular Member*

요 약

재귀적 구조의 SPCPC (single parity check product code)인 CAMC (constant amplitude multi-code) 는 반복적 복호를 행할 때 SPCPC에 비하여 오류 정정 성능이 우수하다. 본 논문에서는 대역확산 신호인 CAMC의 처리 이득이 성능 향상에 미치는 영향을 분석한다. 일반적인 곱 부호에서는 반복적 복호로 오류 정정 과정이 종료되지만, CAMC 는 반복적 복호 후의 역확산 과정에서 추가적으로 오류가 정정된다. 잔존하는 비트 오류의 수가 $(\sqrt{N}/2-1)$개 이하인 경우에는 (N은 코드워드의 길이), 역확산 과정에서 그 오류들은 모두 정정된다. 반복적 복호에서 EI (extrinsic information)의 분포 형태를 관찰한 결과, 초기의 EI 분포는 대체로 랜덤하나, 몇 회의 iteration 후에는 ($-E_{max}$) 혹은 ($+E_{max}$)의 이진 값으로 수렴한다. EI의 분포가 오류 정정의 진행 사항을 반영하는 점을 이용하는 iteration 제어 방법을 실험한 결과 Eb/No 에서 약 0.2 dB의 이득을 얻었다.

Key Words : SPCPC, Processing Gain, Multi-Code, Iterative Decoding, Product Code

ABSTRACT

CAMC (constant amplitude multi-code) has a better performance of error correction in iterative decoding than SPCPC (single parity check product code). CAMC benefits from a processing gain since it belongs to a spread spectrum signal. We show that the processing gain enhances the performance of CAMC. Additional correction of bit errors is achieved in the de-spreading of iteratively decoded signal. If the number of errors which survived the iterative decoding is less than or equal to $(\sqrt{N}/2-1)$, all of the bit errors are removed after the de-spreading. We also propose a stopping criterion in the iterative decoding, which is based on the histogram of EI (extrinsic information). The initial values of EI are randomly distributed, and then they converge to ($-E_{max}$) or ($+E_{max}$) over the iterations. The strength of the convergence reflects how successfully error correction process is performed. Experimental results show that the proposed method achieves a gain of 0.2 dB in Eb/No.

Ⅰ. 서 론

SPCPC (single parity check product code) 부호의 코드워드에서는 데이터 비트들이 다차원 정면체의 형태로 정렬된다. 그림 1은 3-D (3차원) SPCPC의 예를 나타낸 것이다[1]. Q차원 SPCPC는 n 개의 정보 비트에 하나의 패리티 비트를 더하는 방식으로, $(n-1)^Q$개의 정보 비트를 한 변이 $(n-1)$ 비트

※ 본 논문의 연구는 2009년 서울시립대학교 연구교수 지원사업으로 이루어졌습니다
* 서울시립대학교 전자전기컴퓨터공학부 (swchon@uos.ac.kr, yckim@uos.ac.kr)
논문번호 : KICS2010-06-256, 접수일자 : 2010년 6월 7일, 최종논문접수일자 : 2010년 8월 23일

인 정다면체로 전환한 후에 매 차원의 $(n-1)$ 비트 열마다 하나의 패리티를 부가한다. 결과적으로 n^Q 개의 비트로 이루어진 Q차원 정면체인 SPCPC의 코드율은 $((n-1)/n)^Q$이다.

곱 부호 (product code)인 SPCPC에 반복적 복호를 적용하는 과정은 크로스워드 퍼즐을 푸는 것과 유사하다[2,9]. 크로스워드 퍼즐에서 가로와 세로 방향 정보를 모두 이용하는 것과 유사하게 곱 부호의 복호화 과정에서도 수신 심볼을 판정할 때 두 가지 정보를 사용한다. 하나는 수신된 심볼 자체의 값이며, 다른 하나는 코드워드 내에서 그 심볼과 패리티로 연관된 다른 심볼들이 그 심볼에 대해 기대하고 있는 값인 EI(외래 정보: extrinsic information)로서 수신 심볼의 대수 우도비 (log likelihood ratio)로부터 얻어진다. EI의 값은 대응 심볼의 값과 일치하도록 되어 있지만, 오류가 발생하여 수신 심볼 중에 일부가 변하면 일치하지 않을 수도 있다. 여러 iteration 동안 EI와 사전확률정보 (*a priori* information) 가 업데이트되는 relaxation을 거쳐 이러한 불일치성은 감소하면서, 어느 단계 후 더 이상 개선이 이루어지지 않으면 iteration 을 종료한다.

Hagenauer 는 다차원 곱 부호에 적용할 수 있는 SISO (soft input-soft output) 복호 알고리듬을 개발하였고[3], Rankin은 이에 기반한 SPCPC의 반복적 복호법을 제안하였다[1]. Kim은 일정진폭 멀티코드의 일종인 CAMC (constant amplitude multi-code)를 제안하였는데[4], 이는 재귀적인 구조의 SPCPC로서, 그 코드율은 n=4인 SPCPC의 부호율과 같다.

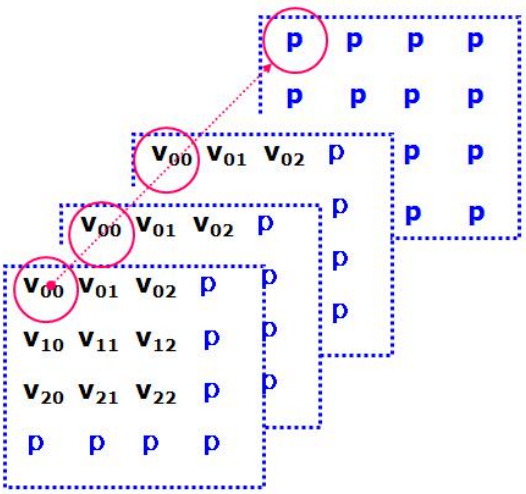


그림 1. 3-D SPCPC의 구조

반복적 복호법을 CAMC에 적용한 결과 CAMC의 오류 정정 성능은 SPCPC에 비하여 1.3 ~ 1.4 dB 우수하며 거의 Shannon의 성능 한계에 가까운 것으로 나타났다[5].

이 논문에서는 CAMC의 처리 이득이 성능 향상에 미치는 영향에 초점을 맞추어 분석한다. 먼저, CAMC는 반복적 복호 후에도 역확산 과정에서 $(n/(n-1))^Q$의 크기의 처리 이득의 효과로 인해 추가적으로 오류가 정정되는 것을 보였다. 일반적인 곱 부호의 경우에는 반복적 복호로 오류 정정 과정이 종료하지만, CAMC의 경우에는 정보 비트와 패리티 비트를 분리하기 위하여 반복적 복호 후에 다시 역확산을 행한다. 이 때 반복적 복호 후에도 코드워드에 잔존하는 비트 오류의 수가 $(\sqrt{N}/2-1)$이하인 경우에는 그 오류들은 역확산 후에 모두 제거된다는 것을 보였다.

또한, 반복적 복호에서 EI의 분포 형태가 오류 정정의 진행 상태를 반영하는 점을 이용하여 iteration을 제어하는 방법을 제안하였다. 반복적 복호에서는 iteration의 실행을 제어하는 stopping criterion이 필요한데, 본 논문에서는 CAMC에서의 EI의 분포 형태로부터 iteration을 제어하는 방법을 제시하고 이로부터 약 0.2 dB의 Eb/No 이득을 얻은 결과를 보였다.

이 논문의 구성은 다음과 같다. 제 2장에서 CAMC의 구조를 간단히 소개하고 제 3장에서 SPCPC와 CAMC의 반복적 복호 과정을 비교한 후, 역확산에 의한 추가적 오류 정정을 서술한다. 제 4장에서는 iteration 이 진행됨에 따라 EI의 분포 형태가 변하는 것을 서술하고, EI의 분포 형태를 성능

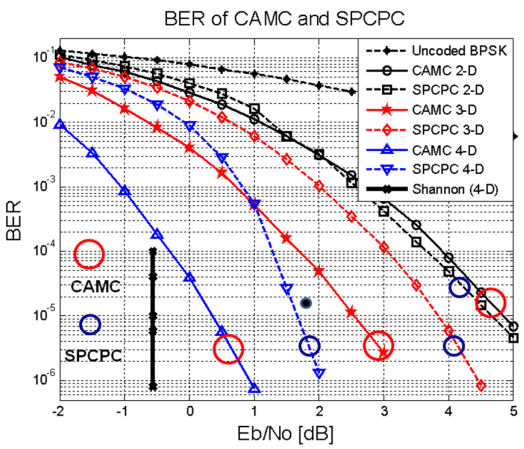


그림 2. CAMC와 SPCPC의 성능 비교

예측자로 이용한 실험 결과를 보인 후, 제 5장에서 결론을 맺는다.

Ⅱ. CAMC의 구성

그림 3은 CAMC 신호 벡터의 부호화부를 나타내며, 신호 벡터는 굵은 글씨로 표기되어 있다[4]. 각 벡터의 윗 첨자는 그 신호 벡터의 길이를 나타내며, {0,1,2,3} 중의 하나로 되어있는 아래 첨자는 4 개의 quadrant 벡터 중 몇 번째인지를 나타내는 인덱스이다. *M* 비트의 입력 비트열은 3 개의 비트로 이루어져 있는 *M*/3 개의 그룹으로 나누어진다. 세 개의 $\mathbf{Q}^{N/4}$ 부호화 과정에서 각각 *N*/4 비트의 quadrant 벡터가 출력되고 이들의 bit-by-bit 패리티 벡터 (*N*/4 비트 크기)를 얻은 후 이들을 모두 일렬로 이어서 *N* 비트 크기의 벡터를 생성된다. 최종적으로 이 *N* 비트 벡터는 $N \times N$ pseudo-Hadamard 행렬, $\widetilde{\mathbf{H}}^{N}$에 의해 확산되어 일정진폭을 갖는 *N* 비트의 벡터, $[v_0, v_1, ..., v_{N-1}]$가 생성된다. 최상위 단계의 CAMC 벡터의 패리티 생성과 확산 과정을 수식으로 표현하면 식 (1) ~ (3)과 같다.

$$\mathbf{v}_3^{N/4} = \overline{\mathbf{v}_0^{N/4} \oplus \mathbf{v}_1^{N/4} \oplus \mathbf{v}_2^{N/4}} \quad \text{(bit by bit)} \tag{1}$$

$$\mathbf{v}^{N} = \frac{1}{2} \cdot \left[\mathbf{v}_0^{N/4} | \mathbf{v}_1^{N/4} | \mathbf{v}_2^{N/4} | \mathbf{v}_3^{N/4}\right] \cdot \widetilde{\mathbf{H}}^{4} \tag{2}$$

$$\widetilde{\mathbf{H}}^{N} = \begin{bmatrix} \mathbf{I}^{N/4} & \mathbf{I}^{N/4} & \mathbf{I}^{N/4} & \mathbf{I}^{N/4} \\ \mathbf{I}^{N/4} & -\mathbf{I}^{N/4} & \mathbf{I}^{N/4} & -\mathbf{I}^{N/4} \\ \mathbf{I}^{N/4} & \mathbf{I}^{N/4} & -\mathbf{I}^{N/4} & -\mathbf{I}^{N/4} \\ \mathbf{I}^{N/4} & -\mathbf{I}^{N/4} & -\mathbf{I}^{N/4} & \mathbf{I}^{N/4} \end{bmatrix} \tag{3}$$

$\widetilde{\mathbf{H}}^{4}$은 $N \times N$ pseudo-Hadamard 행렬이며, $\mathbf{I}^{N/4}$은 $N/4 \times N/4$ Identity 행렬이다. N=4일 경우에는 $\widetilde{\mathbf{H}}^{4}$는 정규 Hadamard 행렬인 $\mathbf{H}^{4}$와 동일하다.

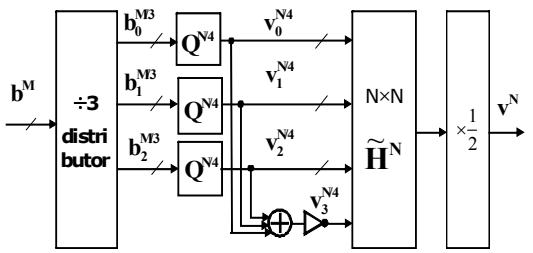


그림 3. *N*-bit CAMC 벡터의 부호화 과정

Ⅲ. 반복적 복호후의 역확산에 의한 추가적 오류 정정

CAMC는 다음과 같은 이유로 인해 SPCPC에 비해 성능이 우수하다. 첫째로 CAMC에는 낮은 무게값을 갖는 코드워드가 없다. 코드의 길이가 N 일 때 그 무게는 $(N \pm \sqrt{N})/2$으로 고정되어 있기 때문이다[10]. 둘째로는 SPCPC와는 달리 CAMC에는 대역확산의 처리 이득이 있기 때문이다[5]. CAMC는 pseudo-Hadamard 행렬로 확산한 신호이며, 복호 과정의 iteration 마다 EI와 사전확률정보는 확산과 역확산을 반복하며, 반복적 복호를 마친 신호는 최종 역확산된 후, 정보 비트와 패리티 비트가 분리된다. CAMC는 이러한 처리 이득의 효과로 인하여 SPCPC에 비하여 우수한 오류 정정 기능을 갖는다. 본 장에서는 이와 같은 역확산에 의한 추가적 오류 정정에 대해 기술한다.

3.1 SPCPC의 반복적 복호

SPCPC의 *q*-번째 차원의 *k*-번째 비트에 대한 LLR, $L_q(X_k)$는 다른 차원의 신호 벡터들과 EI를 교환하면서 순차적으로 값을 업데이트한다. LLR 은 다음과 같이 세 가지 요소로 이루어진다 : 첫째는 수신 신호의 세기이며, 둘째는 사전확률정보 (*a priori* information), (API, $A_q(X_k)$) 이며, 셋째는 외래정보 (extrinsic information (EI, $E_q(X_k)$)이다.

$\mathbf{X} = [X_0, X_1, ..., X_{N-1}], \mathbf{Y} = [Y_0, Y_1, ..., Y_{N-1}]$는 각각 입력신호벡터와 수신신호벡터를 나타낸다[1].

$$L_q(X_k) = \log \frac{\Pr\{X_k = +1 | \mathbf{Y}\}}{\Pr\{X_k = -1 | \mathbf{Y}\}} = \frac{2}{\sigma^2} Y_k + E_q(X_k) + A_q(X_k) \tag{4}$$

$$E_q(X_k) = 2 \tanh^{-1}\left(\prod_{j=0, j \neq k}^{N-1} \tanh\left(\frac{A_q(X_j) + 2/\sigma^2 Y_j}{2} \right) \right) \tag{5}$$

$$A_q(X_k) = \sum_{i=1, i \neq q}^{Q} E_i(X_k) \tag{6}$$

3.2 CAMC의 반복적 복호

CAMC 신호의 패리티 구조는 SPCPC와는 다르다. CAMC에서는 직접적 패리티 관계가 같은 차원 내의 4 개의 quadrant 벡터 사이에서만 존재하므로, 다른 차원의 CAMC와의 패리티 관계를 구하려면

그 벡터를 더 높은 차원으로 확산시키거나 더 낮은 차원으로 역확산시키는 과정을 거쳐야 한다. 패리티와 마찬가지로, EI와 API도 그 자신의 차원에서만 유효한 값이므로, EI나 API를 다른 차원과 교환하기 위해서는 확산과 역확산을 거쳐 다른 차원의 벡터들의 구조와 일치하도록 하여야 한다.

LLR ($L_q(X_k^q)$), API ($A_q(X_k^q)$) 과 EI ($E_q(X_k^q)$) 를 구하는 과정은 식 (7) ~ (9)과 같다. X_k^q는 q-차원으로 재구성된 CAMC 벡터의 k-번째 비트이며, $[X_0^q, X_1^q, ..., X_{L-1}^q]$와 $[Y_0^q, Y_1^q, ..., Y_{L-1}^q]$, $((L=4^q))$는 각각 q-차원으로 재구성된 CAMC 벡터의 입력 벡터와 수신 벡터이다. X_k^Q의 값은 최상위 단계의 LLR을 하드리미팅한 후 이진화하여 얻는다[5].

$$L_q(X_k^q) = \frac{2}{\sigma^2} Y_k^q + E_q(X_k^q) + A_q(X_k^q) \tag{7}$$

$$E_q(X_k^q) = 2\tanh^{-1}\left(\prod_{j=0, j\neq k}^{N-1} \tanh\left(\frac{A_q(X_j^q) + \frac{2}{\sigma^2} Y_j^q}{2} \right) \right) \tag{8}$$

$$A_q(X_k^q) = \sum_{i=1}^{q-1} S(E_i(X_k^q)) + \sum_{i=q+1}^{Q} D(E_i(X_k^q)) \tag{9}$$

3.3 역확산에 의한 추가적 오류 정정

그림 4는 CAMC와 SPCPC의 복호 과정을 도시한 것이다. SPCPC의 경우 반복적 복호 과정의 소프트 출력 값으로부터 직접 정보 비트들의 값을 얻는다. 그러나 CAMC는 반복적 복호 과정만으로는 정보 비트를 얻을 수 없다. CAMC에서는 정보 비트와 패리티 비트가 반복적 복호 과정을 거친 후에도 그대로 섞여 있기 때문이다. 이 둘은 역확산해야 원래대로 분리되므로, CAMC의 복호 과정은 두 단계로 구성된다. 첫 번째 단계는 반복적 복호를 통해 소프트 출력을 얻는 단계이며 두 번째 단계는 그 출력을 $N\times N$ Hadamard 행렬로 역확산하는 과정이다. 반복적 복호 과정에서도 비트 오류의 정정이 이루어지지만, 두 번째 단계인 역확산 과정에서도 추가적인 오류 수정이 이루어진다.

반복적 복호의 소프트 출력은 $N\times N$ 정규 Hadamard 행렬에 의해 역확산 되며 N채널의 상관(correlation) 벡터를 얻을 수 있다. 상관 벡터를 하드리미팅 후 이진 비트로 변환하면 원래의 대응되는 비트 위치로부터 정보 비트 $[b_0, b_1, \cdots, b_{M-1}]$를 추출할 수 있다. 이 때 잡음의 영향이 없다면 각 채널의 상관 값, CF는 $\sqrt{N}$의 크기를 갖게 된다[4,10].

반복적 복호 후 하드리미팅 한 후에도 비트 오류가 남아 있을 수 있으며, 이 때 비트 오류 하나가 있을 때마다, 역확산 후의 상관 벡터의 각 채널의 값은 $\pm\sqrt{N}$로부터 랜덤하게 2씩 증가하거나 감소하게 된다. 잔존 비트 오류의 수가 $\sqrt{N}/2$일 때, 최악의 경우 상관 값의 크기가 $(-\sqrt{N})$에서 계속해서 2씩 증가하여 0이 되거나 $(+\sqrt{N})$에서 2씩 감소하여 0이 될 수도 있다. 그러나 잔존 비트오류의 수가 $N_t = \sqrt{N}/2-1$ 이하인 경우에는, 역확산 후에 상관 값의 부호가 바뀌는 일이 없으므로 하드리미팅한 정보 비트에는 오류가 없다. 따라서 반복적 복호 후의 잔존 비트 오류의 수가 N_t이하일 경우, 역확산

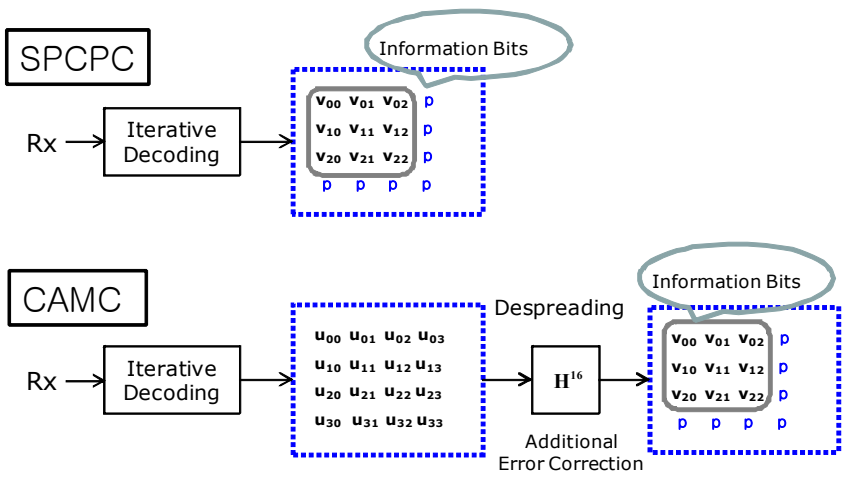


그림 4. CAMC의 복호 과정과 SPCPC 복호 과정의 비교

을 거치면 모든 오류는 제거된다. 표 1 은 이와 같이 정정 가능한 잔존 오류의 수를 코드워드의 길이 N에 따라 정리한 것이다.

표 1. 역확산에 의해 정정 가능한 비트 오류의 수

Code Length	$N=16$	$N=64$	$N=256$	$N=4^D$
CF	4	8	16	$\sqrt{N}$
N_t	1	3	7	$\sqrt{N}/2-1$

Ⅳ. EI의 분포를 이용한 Iteration 제어

본 장에서는 CAMC의 반복적 복호 과정에서의 EI에 대해 분석한다. CAMC는 수신 신호에 대하여 반복적 복호를 수행하고 최종적으로 역확산 과정을 통해 원래의 정보 비트를 추출한다. Iteration 을 거듭하여 오류가 정정되어 감에 따라 EI의 분포가 변하는 형태를 분석하고 EI의 분포로부터 다음 iteration의 효용성을 예측하여 이로써 iteration의 실행 횟수를 제어하여 오류 정정 성능을 높이는 방법을 서술한다.

4.1 EI의 분포

곱 부호의 패리티 비트들은 정보 비트들과 중복된 정보를 갖고 있다. EI는 이러한 중복성을 이용하여 수신 신호의 신호 성분은 강화하고 잡음 성분은 억제하는 역할을 한다. EI 는 반복적으로 업데이트 될 때, 그 EI 값의 분포는 복호 과정의 진행 사항을 반영한다. Berrou는 잡음이 섞여있는 코드워드를 반복적 복호할 때, 처음에는 EI가 랜덤하게 분포되어 있으나, iteration이 거듭됨에 따라 EI의 분포는 0에서 멀어져 양쪽에 대칭적으로 Bi-modal Gaussian 형태로 진행되며, 각 Gaussian의 분산도 점점 작아져서, EI의 값은 (-1)/(+1)로 수렴하는 것을 보였다[2].

이처럼 이진 값으로 수렴하는 이유는 EI는 복호되고 있는 심볼에 대하여 패리티 비트의 관점에서 보는 일종의 기대값이기 때문이다. 잡음이 없는 원래의 코드워드에서는 EI의 값은 대응되는 사전 정보 확률의 값과 동일하게 되어 (-1)이나 (+1)의 값을 갖도록 되어 있다. 잡음이 신호에 섞여 있을 때에는 정보 비트와 패리티 비트 사이의 상응도가 떨어져 심볼 자체의 값과 EI가 기대하는 값 사이의 상관도가 낮아지므로 랜덤한 값을 가질 수 있다.

Iteration이 진행되어 잡음 성분이 억제되면, 상응도가 높아짐에 따라 EI는 이진 값으로 수렴한다. CAMC에서 EI의 분포를 조사하고 이를 SPCPC의 경우와 비교한 결과, iteration 초기의 EI는 랜덤하게 분포되어 있으나, iteration 이 거듭됨에 따라 ($-E_{max}/+E_{max}$)로 수렴하는 것을 확인하였다. $|E_{max}|$는 식 (7) ~ (9) 에서 EI가 너무 큰 값으로 커지지 않도록 제한하는 제한값이다.

그림 5는 3-D CAMC (N=64)의 경우의 EI의 분포를 나타낸 것이다. Eb/No 의 값이 -1.5 dB로 작을 때에는 EI의 값이 특정한 값에 치우치지 않고 0을 중심으로 대체로 균일한 분포를 보이고 있다. 식 (4) ~ (9)에 보인 바와 같이 EI는 최종적으로 수신 신호와 더해져 수신 신호의 부호를 판별하므로, 0에 가까운 EI의 값은 판정에 큰 도움이 되지 않음을 알 수 있다. Eb/No의 값이 증가함에 따라 EI의 값도 ($-E_{max}/+E_{max}$) 쪽으로 이동한다. EI의 값이 작을 때에는 오류 정정에 기여하는 정도가 적으나, 그 값이 커지면 오류가 발생한 원래 수신 심볼의 부호를 바꾸어 바르게 정정할 수도 있다.

그림 6은 CAMC와 비교하기 위하여, Eb/No = 2.5 dB 일 때의 3-D(N=64) SPCPC의 반복적 복호화 과정을 나타낸 것이다. 첫번째 iteration 에서는 EI는 0 근처에 많이 몰려있으나, iteration을 거듭함에 따라 ($-E_{max}/+E_{max}$)쪽으로 수렴되어 간다. 그러나 그 수렴의 세기는 CAMC에 비하여 작다.

그림 7은 3차원 CAMC (N=64)의 경우 Eb/No = -1.5 dB (그림 7a)와 Eb/No = 2.5 dB (그림 7b)일 때, iteration이 1회 ~ 3회 진행되는 동안의 EI의 분포를 나타낸 것이다. Eb/No = -1.5 dB 일 경우에는 초기 EI의 분포는 넓게 퍼져 있으나, iteration

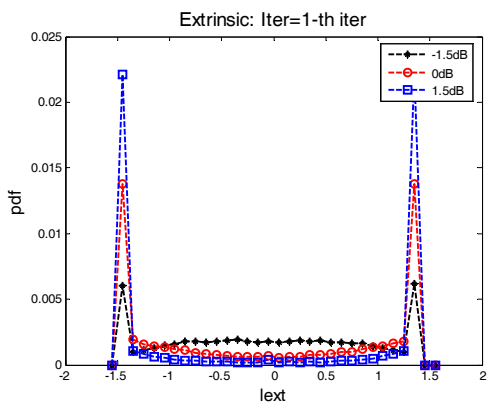


그림 5. CAMC (N=64): Iteration 1 회 후 EI

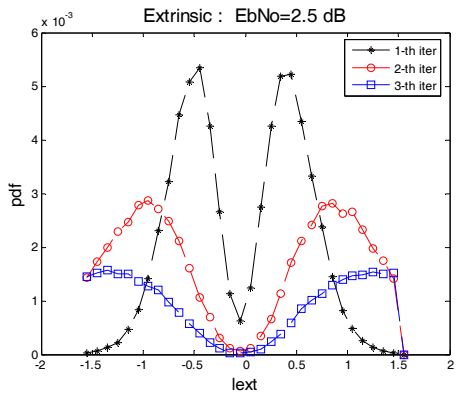


그림 6. SPCPC (N=64)에서의 EI (Eb/No = 2.5 dB)

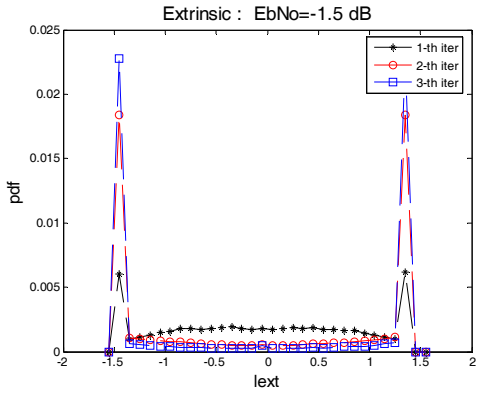


그림 7. (a) CAMC (N=64)의 EI (Eb/No = -1.5 dB)

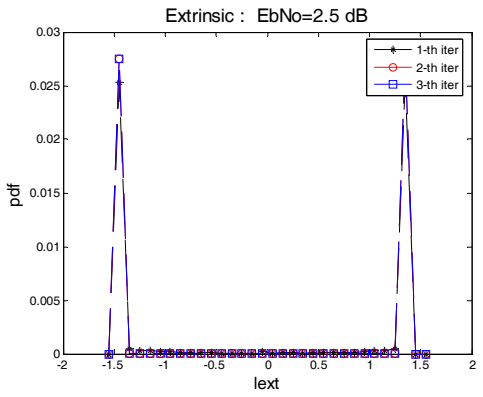


그림 7. (b) CAMC (N=64)의 EI (Eb/No = 2.5 dB)

이 진행됨에 따라 (-E_{max}/+E_{max})쪽으로 수렴한다. 신호대 잡음비가 증가하여 Eb/No = 2.5 dB 인 경우에는 EI의 값들이 처음부터 (-E_{max}/+E_{max})에 강하게 분포되어 있으며, 복호 과정이 진행될수록 이러한 이진화 값으로의 편중은 더욱 분명해진다.

이상과 같은 관찰로부터 다음과 같은 결론을 얻을 수 있다. 첫째, EI가 (-E_{max}/+E_{max})에 집중되어 있는 경우는 오류 정정이 많이 행하여 졌거나, 신호대 잡음비가 커서 코드워드에 비트 오류가 많이 발생하지 않은 경우이다. 둘째, 이와 반대로 EI가 [-E_{max} .. +E_{max}]에 걸쳐 랜덤하게 분포되어 있는 경우는 코드워드에 비교적 많은 수의 비트 오류가 존재하고 있으며 이 오류들의 정정이 성공적이지 않을 가능성이 높다.

4.2 역확산 성능 개선 여부의 예측

그림 5 ~ 7에서 관찰한 결과를 이용하여 다음과 같이 iteration의 실행을 제어하려 한다. EI의 분포 형태는 비트 오류가 정정되는 상태를 반영한다. EI의 절대값이 |E_{max}|로 수렴해나가는 동안은 오류정정이 성공적으로 진행되고 있을 확률이 높으며, 반면에 iteration이 거듭되어도 EI의 절대값이 |E_{max}|로 수렴하지 않고 계속 랜덤한 값들로 남아있는 경우에는 오류 정정이 제대로 진행되고 있지 않을 가능성이 많다. 이러한 특성을 이용하면, EI를 오류정정 성능의 예측자로 활용할 수 있다.

$R_{err}(i)$(i-째 iteration 후 오류의 비율)와 $R_{ei}(i)$(i-째 iteration 후 EI의 절대값이 threshold 이하인 비율)를 관찰한 결과 코드워드의 비트 오류가 정정되어감에 따라 절대값이 threshold 이상인 EI의 비율도 비례적으로 증가한다는 것을 확인하였다. Threshold의 값으로는 0.6 |E_{max}|를 사용하였다.

$$R_{err}(i) = \frac{N_{err}(i)}{N_{err}(first)}, \qquad R_{ei(i)} = \frac{N_{ei}(i)}{N_{ei}(first)} \quad (10)$$

$N_{err}(i)$ = # of erroneous bits in i-th iteration

$N_{EI}(i)$ = # of bits with EI<=threshold in i-th iteration

여러 경우에 대하여 실험한 결과 유사한 결과를 얻었으며, 이를 이용하여 언제 iteration을 멈출지 결정함으로써 오류 정정 성능을 높일 수 있었다. 실제로는 오류의 수를 모르므로 $R_{err}(i)$를 계산할 수 없다. 대신에 $R_{ei}(i)$가 $R_{err}(i)$과 비례하는 특성을 이용하여 $R_{err}(i)$를 사용하는 것이다.

반복적 복호법을 CAMC에 적용하여 BER 성능을 구한 [5]에서의 iteration 횟수는 그 CAMC의 차원의 값과 같도록 일률적으로 정한 것이었다. 즉 3-D CAMC (N=64)의 경우에는 3회, 4-D CAMC (N=256)의 경우에는 4회의 iteration을 행하였다. 본

논문에서는 그림 8에 보인 바와 같은 제어 방법을 사용하였다. 실험 결과 그림 9에 보인 바와 같이 4-D CAMC에서 약 0.2 dB의 Eb/No의 이득을 얻었다. ON은 iteration 제어를 사용한 것이며, OFF는 제어 없이 그 CAMC의 차원의 값을 사용한 것이다. 이 이득의 크기는 0.2 dB 이지만 CAMC의 성능이 Shannon 한계에 가깝다는 점을 고려하면 의미가 있는 값이라고 판단한다.

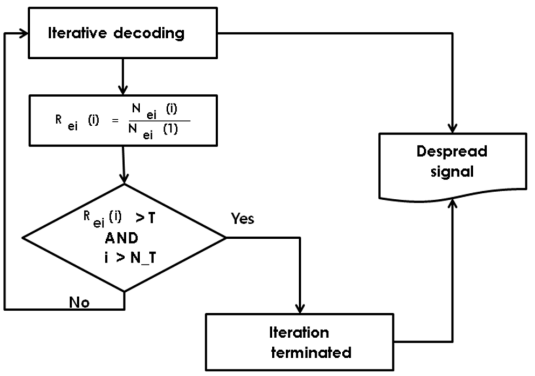


그림 8. EI의 분포를 이용하는 iteration 제어 방법

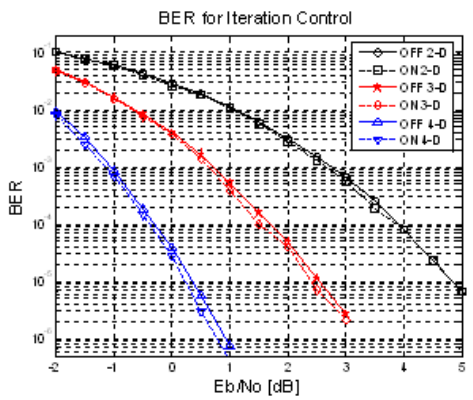


그림 9. Iteration 제어 ON과 OFF 의 성능

V. 결 론

CAMC는 SPCPC와 유사한 구조이나 반복적 복호를 행할 때 SPCPC 보다 성능이 우수하다. CAMC는 무게 분포와 처리 이득 면에서 SPCPC에 비하여 우수하여, 성능이 거의 Shannon 한계에 가깝다. 본 논문에서는 CAMC의 처리 이득이 성능 향상에 미치는 영향을 두 측면에서 분석하였다.

첫째, CAMC에서는 반복적 복호 후에도 역확산 과정에서 추가적으로 오류가 정정되는 것을 보였다. SPCPC와 같은 일반적인 곱 부호의 경우, 오류 정정은 반복적 복호에서 끝나지만, CAMC의 경우에는 반복적 복호 후에 역확산을 행한 후 상관 값의 부호로부터 정보 비트를 추출한다. 이 때 잔존하는 비트 오류의 수가 $N_i = \sqrt{N}/2 - 1$ 이하인 경우에는 역확산을 거친 후의 상관 값의 부호가 원래의 정보 비트의 값의 부호와 동일함을 유지하므로, 잔존하는 비트 오류는 모두 제거된다.

둘째로, EI의 분포 형태로부터 iteration을 제어하는 방법을 제안하였다. 반복적 복호에서 EI 가 초기에는 랜덤하게 분포되어 있지만 오류 정정이 진행될수록 ($-E_{max}$)혹은 ($+E_{max}$)의 이진 값으로 수렴한다. 실험 결과 오류가 정정될수록 threshold 이하의 절대값을 갖는 EI의 비율도 따라서 줄어드는 것을 알 수 있었으며, 이 비율로부터 iteration의 stopping criterion을 도출하였다. 이러한 제어 방법을 이용한 실험 결과 Eb/No 에서 0.2 dB의 이득을 얻었다. CAMC의 성능이 이미 Shannon 한계에 가까운 점을 고려하면 이와 같은 수치는 의미가 있는 값이다.

이러한 연구결과는 저전력 근거리 무선 통신 솔류션의 FEC 기능으로 활용할 수 있다. 그러나 현재의 복호 알고리듬은 그 계산량이 너무 많은 편이다. 이에 대한 해결책으로 오류 정정 기능이 추가되는 칩의 연산부의 Finite Word Length를 고려하는 정수 연산 시뮬레이션을 거쳐 요구되는 성능에 필요한 Word Length를 결정하여 불필요한 계산을 줄이려 한다.

참 고 문 헌

〔1〕 D. Rankin and T. Gulliver, “Single parity check product codes,”IEEE Trans. Comm., vol.49, no.8, pp.1354-1362, Aug. 2001.

〔2〕 C. Berrou and A. Glavieux, “Near Optimum Error Correcting Coding and Decoding : Turbo-Coding,” IEEE Trans. On Communications, Vol.44, No.10, pp.1261-1271, Oct. 1996.

〔3〕 J. Hagenauer, E. Offer and L. Papke, “Iterative decoding of binary block and convolutional codes,” IEEE Trans. Inform. Theory, Vol.42, pp.429-445, Mar. 1974.

〔4〕 Y. Kim, “Recursive generation of constant amplitude multi-code DSCDMA signal,” IEE

Electronics Letters, Vol.39, No.25, pp.1782-1783, Dec. 2003.

〔5〕 Y. Kim, "Constant amplitude multi-code CDMA with built-in single parity check product code," IEEE Communications Letters, vol. 10, No.1, pp.4-6, Jan. 2006.

〔6〕 G. Caire, G. Taricco and G. Battail, "A Weight distribution and performance of the iterated product of single-parity-check codes," Proceed. of IEEE GLOBECOM, pp.206-211, Dec. 1994.

〔7〕 E. Biglieri and V. Volski, "Approximately Gaussian weight distribution of the iterated product of single-parity-check codes," IEE Electronics Letters, Vol.30, No.12, pp.923-924, Jun. 1994.

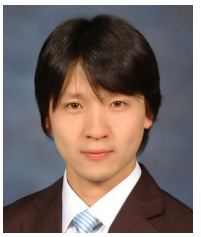
〔8〕 D. Yue and E. Yang, "Aymptotically Gaussian weight distribution and performance of multicomponent turbo block codes and product codes,"IEEE Trans. on Comm, Vol.52, No. 5, pp.728-736, May, 2004.

〔9〕 G. Battail, "A conceptual framework for understanding turbo codes,"IEEE J. Selected Areas Comm., Vol16, No.2 pp.245-254, Feb. 1998.

〔10〕 I. Park, B. Kim, and Y.Kim, "Processing Gain and Fixed Weights of a Recursive Single Parity Check Product Code," Information Journal, Vol.12, No.4, pp.923-932, Jul. 2009.

[11] S. Lin and D. Costello, "Error control coding," 2nd ed. Pearson Prentice Hall, 2004.

전 수 원 (Suwon Chon) 준회원



2008년 원광대학교 전자 전기공학부

2010년 서울시립대학교 전자전기컴퓨터공학부 석사

김 용 철 (Yong Cheol Kim) 정회원

1981년 서울대학교 전자공학과

1983년 KAIST 전기 및 전자공학과 석사

1983년~1986년 금성전기연구소

1993년 University of Southern California 박사

1993~1996년 LG정밀연구소 전문팀장

1996~현재 서울시립대학교 전자전기컴퓨터공학부 교수