International Journal of Multimedia and Ubiquitous Engineering
Vol.11, No.8 (2016), pp.121-130
http://dx.doi.org/10.14257/ijmue.2016.11.8.13

Efficient Decoding Technique for Block Product Turbo Codes

Je-Hun Lim[1] and Young-Joon Song[2]

[1]*Department of intelligent transport systems, 2571 Geoga-daero, Gangseo-gu, Busan, 46770, Korea,*
[2]*Department of Electronic Engineering, Kumoh National Institute of Technology, 1 Yangho-dong, Gumi, Gyungbuk, 730-71, Korea*
[1]*jehun.lim@koinfra.com,* [2]*yjsong@kumoh.ac.kr*

Abstract

In this paper, we propose an efficient technique for an iterative soft input soft output (SISO) decoding algorithm of block turbo codes with constituent Hamming codes. Pyndiah's iterative decoding algorithm for block turbo codes supports various code rates by using concatenation, and shows high bit error rate (BER) performance in an additive white Gaussian noise channel (AWGN) environment as the iterative decoding progresses. However, one disadvantage of this code is that the delay increases during decoding with an increasing number of iterations. By employing syndrome-based iterative SISO decoding and iterative SISO decoding using an alpha value, we could obtain high BER performance. In addition, we propose a SISO-hybrid maximum likelihood decoding (SISO-H-MLD) algorithm which achieves almost the same performance as SISO decoding while reducing the number of iterations.

Keywords*: block turbo codes, Hamming code, hybrid maximum likelihood decoding, SISO*

1. Introduction

Pyndiah's block turbo codes support various code rates by using concatenation and boast of high BER performance, which is close to the Shannon limit in an additive white Gaussian noise (AWGN) channel environment through soft input soft output (SISO) iterative decoding [1-3]. BER performance improves with an increasing number of iterations, however, the complex structure of iterative techniques and the log likelihood ratio (LLR) calculations result in a greater delay [4]. The complexity of calculations can be reduced using the low-complexity SISO algorithm proposed by Benjamin Muller, but redundant iterations accompanying the increasing number of iterations result in a greater delay and less significant improvement in the bit error rate (BER) performance [5].

To resolve the aforementioned problems, Lim and Song proposed a method to halt iterative decoding through a syndrome check [6], and inputted an alpha value into the decoder to enhance BER performance [7]. To improve BER performance while reducing the number of iterations, we propose a SISO-hybrid maximum likelihood decoding (SISO-H-MLD) algorithm, which is a combination of hybrid maximum likelihood decoding [8] and SISO decoding. Further, a performance comparison between the proposed algorithm and the conventional techniques in an AWGN channel environment is presented.

The rest of this paper is organized as follows: Section 2 explains the encoding process of block turbo codes, and Section 3 introduces the low-complexity SISO decoding algorithm proposed by Benjamin Muller and discusses the problems with it. Section 4 describes the existing SISO decoding technique of block turbo codes by using an alpha value, and Section 5 proposes a SISO-H-MLD technique for block turbo codes. A simulation is carried out for a comparative performance analysis between the existing

ISSN: 1975-0080 IJMUE

techniques and the proposed decoding technique. Finally, Section 6 provides the results of the simulation, followed by the conclusion.

2. Encoding for Block Turbo Codes

In this paper, we use the product code for the encoding of block turbo codes, BPTC (196,96), as an example. First proposed by Elias, this product code generates a long code with an error-correction ability by using a two-block code (B_1, B_2) [9]. The encoding of block turbo codes follows the matrix shown in Figure 1 [10], and consists of the following:

Step 1) The information vector M, in the form of a 9×11 matrix, is assigned to the message $I_0, I_1, \cdots, I_{95}$, as shown in Figure 1. The difference between 99 bits and 96 bits is 3 bits, and these are filled by $R(0), R(1), R(2)$ with all zero values.

Step 2) By using (1), the information vector M generates the 9×15 code matrix using the generator matrix G_{15}, where $\cdot$ refers to the modulo-2 multiplication.

$$P_1 = M \cdot G_{15} \tag{1}$$

A 9×4 matrix in the 9×15 matrix P_1 is the parity part of the code.

Step 3) P_1 is transformed into the transposed matrix P_1^T to generate a 15×9 matrix, and P_2 is produced by (2), where G_{13} denotes the 9×13 generator matrix for the shortened Hamming code (13,9), as shown in Figure 3.

$$P_2 = P_1^T \cdot G_{13} \tag{2}$$

The difference between 195 bits and 196 bits is filled by adding R_3=0, corresponding to 1 bit.

Figure 1. Matrix Structure of Block Product Turbo Code (196,96)

1	0	0	0	0	0	0	0	0	0	0	1	0	0	1
0	1	0	0	0	0	0	0	0	0	0	1	1	0	1
0	0	1	0	0	0	0	0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0	0	0	0	1	1	1	0
0	0	0	0	1	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	0	1	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	1	0	0	1	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	1	1

Figure 2. Generator Matrix of Hamming C ode (15,11)

1	0	0	0	0	0	0	0	0	1	1	1	1
0	1	0	0	0	0	0	0	0	1	1	1	0
0	0	1	0	0	0	0	0	0	0	1	1	1
0	0	0	1	0	0	0	0	0	1	0	1	0
0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	1	0	0	0	1	0	1	1
0	0	0	0	0	0	1	0	0	1	1	0	0
0	0	0	0	0	0	0	1	0	0	1	1	0
0	0	0	0	0	0	0	0	1	0	0	1	1

Figure 3. Generator Matrix of Hamming Code (13,9)

3. Low-Complexity SISO Decoding

Block turbo codes are decoded using SISO iterative decoding, which outputs soft values. As the number of iterations increases during decoding, the reliability of the final output LLR increases, thereby enhancing the BER performance. However, one disadvantage of this iterative decoding technique is the greater delay that can be caused by complex calculations with an increasing number of iterations. To solve this problem, in this paper, we introduce a method using the low-complexity SISO decoding [5] proposed by Benjamin Muller. Since the decoding relies on an error table, error syndromes are first calculated through a syndrome check, and then, an error table corresponding to each error syndrome is generated. For Hamming code (15,11), the code length is 15, and the generated error vector v accounts for $2^{15}-1$ cases. The error syndrome S is derived for all error vectors through binary operation using the transposed matrix H_{15}^{T}. For the sake of simplicity, in this paper, we only consider error patterns with less than or equal to three errors. All the possible error syndrome values are equal to the row vectors of H_{15}^{T}. The error table for Hamming code (15,11) with less than four errors is denoted by $T_a = (T_a^1, T_a^2, \cdots T_a^{15})$, where T_a^i represents the error table for the *i*-th syndrome. Completing this process, we have 15 error tables for Hamming code (15,11). Since Hamming code (13,9) is a shortened version of Hamming code (15,11), we produce the error table $T_b = (T_b^1, T_b^2, \cdots T_b^{15})$ for the shortened code.

When error tables T_a, T_b are generated, the signal r received through the AWGN channel is calculated as follows:

$$C = \ln \frac{\exp\left(-\frac{E_s}{N_0}(r-1)^2\right)}{\exp\left(-\frac{E_s}{N_0}(r+1)^2\right)} = \frac{4E_s}{N_0} r \tag{3}$$

The calculated channel value C is soft information, which is converted to $\tilde{d}$ via a hard decision to obtain the error syndrome S. A decision of 0 is made when the polarity of i rows and j columns is positive $(+)$, or 1 when the polarity is negative $(-)$. During the syndrome check, the information is considered reliable if the error syndrome is 0. Otherwise, an error table that matches the error syndrome is searched among the generated error tables.

For instance, let us assume the error syndrome $(1\ 0\ 0\ 1)$ from the syndrome check for the channel value C and the parity check matrix H_{15} of Hamming code (15,11). Since the error syndrome $(1\ 0\ 0\ 1)$ is the same as the first column vector of the parity check H_{15}, we choose the error table T_a^1 from $T_a = (T_a^1, T_a^2, \cdots T_a^{15})$.

$$T_a^1 = \begin{bmatrix} 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,1 \\ \vdots \\ 0\,1\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \end{bmatrix}$$

Where, the number of errors in error table T_a^1 is confined to less than four. After completing this process, we use (4) to obtain the probability P_j for the incoming codeword.

$$P_j = \frac{\exp(|C_j|)}{1 + \exp(|C_j|)} \tag{4}$$

The probability P_j becomes $\dot{P}_i$ when we use the row vector e_i of the error table and (5).

$$\dot{P}_i = \prod_j \begin{cases} P_j & \text{if } e_{i,j} = 0 \\ 1 - P_j & \text{if } e_{i,j} = 1 \end{cases} \tag{5}$$

By summing the calculated probabilities $\dot{P}_i$ and normalizing them to 1 as shown in (6), we obtain the normalized probability P_i. P_i denotes the probability of the input channel value having the error pattern of e_i.

$$P_i = \frac{\dot{P}_i}{\sum_i \dot{P}_i} \tag{6}$$

For the SISO iterative decoding, the LLR value for the j-th code bit is given by

$$\mathrm{LLR} = \ln \frac{\hat{P}(q_j = +1|r)}{\hat{P}(q_j = -1|r)} \tag{7}$$

where

$$\hat{P}(q_j = +|r) = \sum_{e_{i,j} = \tilde{d}_j} P_i \tag{8}$$

$$\hat{P}(q_j = -1|r) = 1 - \hat{P}(q_j = +1|r) \tag{9}$$

When the low-complexity SISO algorithm is applied to the block turbo code, iterative decoding is carried out by the two decoders connected in series with the deinterleaver and

the interleaver. A syndrome check is performed on the received information signal entered into the decoders. The input information becomes the estimated codeword if the error syndrome is zero, which implies that the data entered into the two decoders are error-free. There can be some other cases of the zero syndromes as the number of iterations increases. This case can cause the absolute value of LLR to become so high that the decoders may not function properly when the $(\pm\infty)$ values are entered with the inner and outer decoders operating on each other. This is a serious problem with the syndrome-based SISO turbo decoder. To prevent $(\pm\infty)$ from occurring in the final output values, an optimal value in the range of 1-2 is applied, as verified in the simulations of the previous research result [6].

4. Iterative SISO Decoding Using an Alpha Value

In order to enhance the performance of the technique developed from Benjamin Muller's method [5], this paper introduces an iterative decoding technique capable of achieving outstanding BER performance using an alpha obtained from simulation on turbo decoder [7]. Figure 4 shows the structure of iterative decoding produced by adding an alpha value retrieved from simulation on turbo decoder connected in series [11-12]. Here, α is (2, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9) as E_b/N_0 increases from 1dB to 10dB at 1dB interval. The SISO algorithm of turbo codes using an alpha value consists of the six steps outlined below.

Step 1) Since the output decoder has no output value during the first iterative decoding, SISO decoding is performed with C as the input channel value in the inner decoder. When decoding is completed, a reliable LLR_1 is given as the final output of the decoder.

The output value of the decoder is derived using the improved syndrome-based iterative SISO decoding algorithm, previously described in Section 3.

Step 2) The deinterleaved $\pi^{-1}(LLR_1)$ is multiplied with α obtained from simulations and it is entered into the outer decoder.

Step 3) The input $(LLR_1 \times \alpha)$ leads to the output LLR_2 through SISO decoding. The output LLR_2 is calculated with deinterleaved $\pi^{-1}(LLR_1)$, and the first iterative decoding ends.

Step 4) During the second iterative decoding, the value obtained in Step 3 is interleaved by $\pi[LLR_2 - \pi^{-1}(LLR_1)]$ and multiplied with α, and then, it is entered into the inner decoder together with the channel value C. The input values give a new output value of N_LLR_1 through the SISO decoder.

Step 5) The output N_LLR_1 is calculated with $\pi[LLR_2 - \pi^{-1}(LLR_1)]$, deinterleaved by $\pi^{-1}[N\_LLR_1 - \pi[LLR_2 - \pi^{-1}(LLR_1)]]$ and multiplied by α before being entered into the outer decoder.

Step 6) The value entered into the outer decoder provides a new N_LLR_2 once SISO decoding is completed. The third iterative decoding and beyond follows the same procedure as the second.

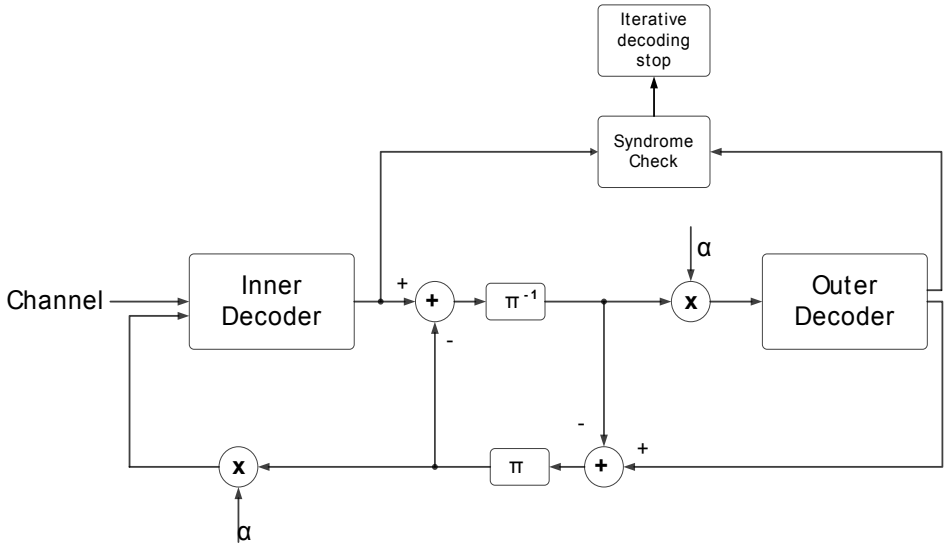

Figure 4. Serial Structure of Turbo Decoder

A greater delay is caused by redundant iterations accompanying the increasing number of iterations. A syndrome check is performed on all the rows and columns of the output values from the inner and the outer decoders, and the problem is resolved by stopping iterative decoding when the syndrome is 0.

5. SISO-Hybrid Maximum Likelihood Decoding

It is true that decoding must be performed with a high number of iterations in order to attain high BER performance. However, doing so results in a greater delay because of the increased complexity of the calculations. In this paper, we propose a SISO-H-MLD algorithm that achieves outstanding BER performance while reducing the number of iterations. This algorithm first performs a syndrome check on all the rows and columns of the output LLR value from the outer decoder by using alpha values [7], and then, applies hybrid maximum likelihood decoding [8] when the error syndrome is not 0.

The proposed decoding technique is explained below with BPTC code (196,96) as an example.

Step 1) The output LLR of the outer decoder converts the 13×15 matrix into its transposed 15×13 form.

$$X = \begin{bmatrix} x_{1,1}, x_{1,2}, \cdots, x_{1,12}, x_{1,13} \\ x_{2,1}, x_{2,2}, \cdots, x_{2,12}, x_{2,13} \\ \vdots \\ x_{15,1}, x_{15,2}, \cdots, x_{15,12}, x_{15,13} \end{bmatrix}$$

Step 2) Between Hamming code (15,11) and Hamming code (13,9), Hamming code (13,9) has a shorter code length. Its generator matrix G_{13} is classified into G_a and G_b.

$$G_a = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix}$$

$$G_b = \begin{bmatrix} g_5 \\ g_6 \\ g_7 \\ g_8 \end{bmatrix}$$

Step 3) Let Z_a consist of the 2^5 binary vectors corresponding to the generator matrix G_a. Then the generator matrix G_a produces the candidate codeword W_a by using (10) and a modulo 2 operation. Similarly, the generator matrix G_b produces the candidate code vector W_b by using (11).

$$W_a = \begin{bmatrix} w_a^{(1)} \\ w_a^{(2)} \\ \vdots \\ w_a^{(2^5)} \end{bmatrix} = \begin{bmatrix} z_a^{(1)} \\ z_a^{(2)} \\ \vdots \\ z_a^{(2^5)} \end{bmatrix} \cdot \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \end{bmatrix} \tag{10}$$

$$W_b = \begin{bmatrix} w_b^{(1)} \\ w_b^{(2)} \\ \vdots \\ w_b^{(2^4)} \end{bmatrix} = \begin{bmatrix} z_b^{(1)} \\ z_b^{(2)} \\ \vdots \\ z_b^{(2^4)} \end{bmatrix} \cdot \begin{bmatrix} g_5 \\ g_6 \\ g_7 \\ g_8 \end{bmatrix} \tag{11}$$

Step 4) Let $\boldsymbol{x}$ be a row vector of X. Further, (12) is executed to produce $f^{(n)}$, where $n = 1 \sim 2^4$.

$$f^{(n)} = [x] .* \left[(-1)^{W_{b_1}^{(n)}}, (-1)^{W_{b_2}^{(n)}}, \cdots, (-1)^{W_{b_{15}}^{(n)}} \right] \tag{12}$$

Step 5) The candidate codeword $W_a^{(k)}$ is calculated with $f^{(n)}$ as (13), and the result $u^{(k)}$ is stored in the memory buffer U, where $k = 1 \sim 2^5$.

$$u^{(k)} = \left[(-1)^{W_{a_1}^{(k)}}, (-1)^{W_{a_2}^{(k)}}, \cdots, (-1)^{W_{a_{15}}^{(k)}} \right] .* \begin{bmatrix} f^{(1)} \\ f^{(2)} \\ \vdots \\ f^{(2^4)} \end{bmatrix} \tag{13}$$

where

$$U = \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(2^5)} \end{bmatrix}$$

Step 6) If the maximum value of U is selected, then two binary vectors, $z_a^{(i)}$ and $z_b^{(j)}$, corresponding to the location index of the maximum value form the decoded message vector $\boldsymbol{m} = (z_a^{(i)}, z_b^{(j)})$ for the $\boldsymbol{x}$ of X.

Step 7) This process is repeated for all vectors of X.

After the above process is completed, the decoded LLR takes on the form of 15×9 and we obtain the 11×9 message symbol from the block turbo code.

5.1. Performance of the Proposed Algorithm

In this paper, we compared the performance of the conventional alpha-based SISO iterative decoding algorithm with that of the proposed SISO-H-MLD, as shown in Figure 5. The figure shows a comparison of the BER performance of the existing technique with the number of iterations ranging from 1 to 5 and that of the proposed technique with one iteration in an AWGN channel environment. The result shows that SISO-H-MLD shows almost the same performance gain as the iterative SISO decoding algorithm using an alpha value after four iterations.

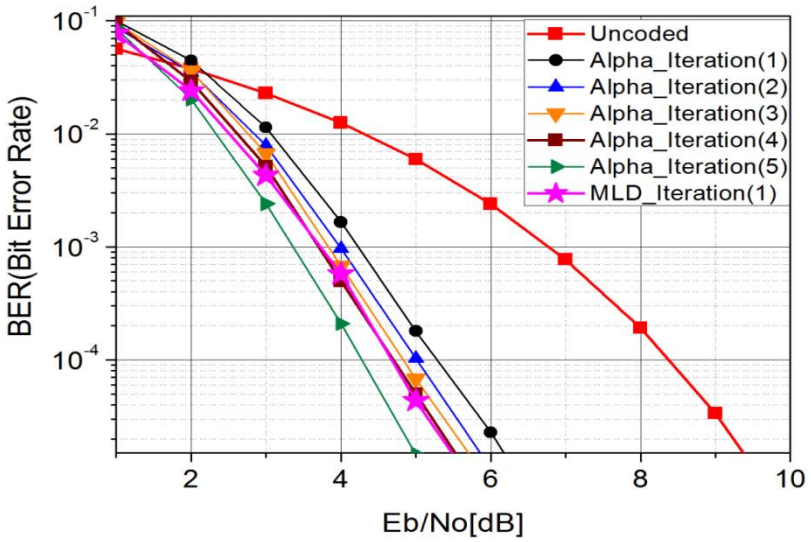

Figure 5. Performance of Block Product Turbo Code (196,96)

6. Conclusion

In this paper we have proposed an efficient SISO-H-MLD algorithm for block turbo codes. This algorithm can obtain the higher BER performance than the conventional iterative SISO decoding algorithm using an alpha value while reducing the number of iterations. To show the performance of the proposed decoding algorithm, we demonstrated the decoding of block product turbo codes in an AWGN channel environment, which shows almost the same performance gain as the iterative SISO decoding algorithm using an alpha value in four iterations.

Acknowledgment

This paper was supported by Kumoh National Institute of Technology.

References

[1] R. M Pyndiah, A. Glavieux, A. Picart and S. Jacq, "Near Optimum Decoding of Products Codes", in Proceeding of IEEE GLOBECOM 94 Conference, vol. 1/3, **(1994)**, pp. 1064-1071.
[2] R. M. Pyndiah, "Near Optimum Decoding of Products Codes: Block Turbo Codes", IEEE Transaction on. Communication, vol. 46, no. 8, **(1998)**, pp. 1003-1010.
[3] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting and Decoding: Turbo Codes", Proceeding IEEE International Conf. Commun (ICC93), **(1993)**, pp. 1064-1070.
[4] S. Pietrobon, "Implementation and Performance of a Turbo/MAP Decoder", Int. J. Satellite comm., vol. 16, **(1998)**, pp. 23-46.
[5] B. Muller, M. Holters and U. Zolzer, "Low Complexity Soft-Input Soft-Output Hamming Decoder", FITCE Congress, **(2011)**, pp. 1-5.
[6] J. H. Lim, S. P. Nah and Y. J. Song, "Syndrome Based Soft-Input Soft-Output Decoding of BPTC (196,96) with Performance Analysis", Journal of KIIT, vol. 12, no. 11, **(2014)**, pp. 111-117.
[7] J. H. Lim and Y. J. Song, "Performance Improvement Technique for SISO Decoding of Block Turbo Code with Hamming Constituent Codes", Journal of KIIT, vol. 13, no. 5, **(2015)**, pp. 19-26.
[8] Y. J. Song, "Hybrid Maximum Likelihood Decoding for Linear Block Codes", International Journal of Multimedia & Ubiquitous Engineering, vol. 9, no. 10, **(2014)**, pp. 91-100.
[9] P. Elias, "Error-Free Coding", Transactions of the IRE Professional Group on Information Theory, vol. IT-4, **(1970)**, pp. 29-37.
[10] Y. H. Chen, J. H. Hsiao, K. F. Lin and C. C. Yeh, "BPTC(196,96) Code Simulation and Implementation in C-code DPS Chip", Consumer Electronics, Communications and Networks(CECNet), **(2011)**, pp. 5-10.
[11] T. K. Moon, "Error Correction Coding", John Wiley & Sons, **(2005)**, pp. 581-632.

[12] S. Benedetto and G. Montorsi, "Serial Concatenation of Interleaved Codes: Performance Analysis", IEEE Transactions on Communication, vol. 44, no. 3, **(1996)**, pp. 909-926.
[13] J. H. Lim, J. H. Lee, M. S. Shin, G. H. Cho and Y. J. Song, "Low-Complexity and High-Performance SISO Decoding for Block Product Turbo Code (105,44)", 2015 8th International Conference on Control and Automation (CA2015) **(2015)**, pp. 13-16.

Authors

Je Hun Lim, received his BS an MS in Electronic Engineering from Kumoh National Institute of Technology, Korea, in 2014 and 2016 respectively. Since May 2016, He has been working at the Korea Infra Management Inc., Busan, Korea. His research interests include channel coding, mobile/multimedia communication and intelligent transport systems.

Young Joon Song, received his BS, MS, and PhD in Electronic Communications Engineering from Hanyang University, Korea, in 1987, 1994, and 1999, respectively.

He was a principal research engineer with LG Electronics Inc. from 1994 to 2002. Since 2002, he has been with the Department of Electronic Engineering, Kumoh National Institute of Technology, Gumi, Korea, where he is currently a Professor. During 2006, he was at the University of Hawaii at Manoa, as a visiting scholar. His research interests include channel coding and mobile/multimedia communication systems.