Application of Shaping Technique to Multi-level Turbo-coded Modulation

Amir K. Khandani† and W. Tong††
†Coding and Signal Transmission Laboratory (www.cst.uwaterloo.ca)
E&CE Dept., Univ. of Waterloo, 200 University Ave. West, Waterloo, ON, Canada, N2L 3G1
††Nortel Networks, 100 Constellation Crescent, Nepean, ON, Canada, K2G 6J8

Abstract— **This article presents a method for the combined Turbo-code and QAM modulation using coset coding. A Turbo-code (acting on non-binary symbols) is used to select a coset along each one-D co-ordinate using a one-D lattice partition (Ungerboeck partition). The proposed method satisfies the conditions for the separability of shaping and coding, and consequently, can be easily integrated with a shaping algorithm. Numerical results are presented which show that the proposed configuration outperforms the best known techniques in terms of the complexity, delay and BER performance (even ignoring the extra gain achievable through shaping). Furthermore, we are able to achieve an additional gain of about 1 dB through integration with shaping. The proposed method also demonstrates an excellent performance near the error floor region.**

I. Introduction

Coded modulation goes back to 1974, when Massey suggested the notion of improving system performance by looking at coding and modulation as a combined entity [1]. The most successful step towards implementing Massey's thoughts was the invention of *Trellis-Coded Modulation* (TCM) by Ungerboeck [2]. A distinct feature of TCM is the application of labeling by set partitioning where the constellation is successively binary partitioned into congruent subsets defining a mapping of binary addresses to the signal points. Almost at the same time and independent of [2], Imai and Hirakawa [3] proposed *Block-Coded Modulation* (BCM) using multi-level binary coding. This is based on using a set of parallel binary block codes to specify different bit values in a given binary labeling of the constellation points. The basic structure proposed in [3] has been further studied and generalized in a number of subsequent research works. Reference [4] which is the state of the art article in this category includes a detailed review of the relevant literature.

Shaping concerns the selection of the boundary of a multi-dimensional signal constellation to reduce its average energy. It is well known that by using shaping techniques in conjunction with conventional band-width efficient modulation methods, one can achieve an additional shaping gain (refer to [5] for definition) of about one dB with a modest increase in the complexity. In the work of Wei, [6], shaping is a side effect of the method employed to transmit a non-integral number of bits per two dimensions. Forney and Wei generalized this method in [5]. Conway and Sloane in [7] introduced the idea of the Voronoi constellation based on using the Voronoi region of a lattice Λ_s as the shaping region. Voronoi constellations are further studied by Forney in [8]. In [9], Calderbank and Ozarow introduced a shaping method in which the 2-D subspaces are partitioned into equal sized shells of increasing average energy where a shaping code is used to specify the sequence of the 2-D shells. Lang and Longstaff in [10] use an addressing scheme which first divides the final constellation into energy shells. Then, a point in a shell is found by successively decomposing the space into lower-dimensional subspaces via generating function techniques. The addressing scheme of Lang and Longstaff is further discussed and elaborated in [11][12][13][14][15]. The idea of the trellis shaping is introduced in [16]. This is based on using an infinite dimensional Voronoi region, determined by a convolutional code, to shape the constellation. In [17], Kschischang and Pasupathy discuss a shaping method which is based on using the 2-D points with non-equal probability. In [13], [18], [19] some very low complexity addressing schemes are given which achieve (or closely approximate) points on the optimum trade-off curves. The method reported in [18] has been the lowest complexity addressing method for spaces of dimensionality up to 32, and the the method reported in [19] has been the lowest complexity addressing method for spaces of higher dimensionality (up to 512). In the current article, we present an addressing scheme based on some improvements upon the method proposed in [18] for addressing in a 32 dimensional shaped constellation which results in a lower complexity as compared to all the methods reported in the literature.

In 1993, a new class of channel codes, called Turbo-codes, were announced which have an astonishing performance with a reasonable complexity [20]. The basic idea of Turbo-codes is to make use of some convolutional component codes which are connected in parallel through pseudo-random interleavers. There have been a number of successful research works addressing the problem of band-width efficient modulation in conjunction with Turbo-codes [21][22][23][24][25][26][27][28][29][30][31].

This article presents methods for the combined Turbo-code and QAM modulation using coset coding where the underlying binary code is a symbol-based Turbo-code. The proposed method satisfies the conditions for the separability of shaping and coding in [5], and consequently, can be easily integrated with a shaping algorithm.

References [22][23][26][27][31] present structures based on parallel concatenation of trellis-coded modulation (TCM) schemes using binary component codes. In [22][23], the interleaver operates on bits, while in [26][27][31] the interleaver operates on symbols. The use of symbol interleaving has the disadvantage of reducing the effective interleaving gain, however, it also offers some advantages in terms of complexity and performance [33][34][36][37][38] which usually results in a net gain

This work is financially supported by Nortel Networks.

for symbol vs. bit interleaving. Note that the trellis branching factor in [26][27][31][33][34] depends on the number of constellation points and consequently the complexity of the soft output decoder for the underlying trellis can be substantially higher than the standard binary Turbo code. The method proposed here is also based on symbol interleaving, however, its underlying binary component codes operate on the cosets (and not on the individual constellation points), and consequently, the decoding complexity does not grow with the size of the constellation. Reference [32] discuses the application of shaping technique to Turbo-codes using a nonuniform constellation. However, the shaping gain achievable using the method proposed in [32] is limited to about 0.2 db. The problem of shaping in conjunction with Turbo-codes is also briefly mentioned in [4].

A common approach to Turbo-coded modulation is based on a direct mapping of the output bits of a binary Turbo-code to the points of a base constellation via an interleaver. In this configuration, the channel can be considered as m binary input, continuous output channels (the output of each such binary channel is the corresponding bit LLR), where 2^m is the number of points in the base constellation. These binary channels are not necessarily symmetrical and also their outputs are dependent on each other. The value of the achievable rate over these equivalent binary channels (value of the mutual information associated with bits) can be easily computed. Figure 1 shows different configurations for the overall achievable rate of a 4-point one-dimensional constellation. This figure also contains the capacity of an AWGN channel for the sake of comparison. The two curves labeled as "binary independent channels" are simply obtained by adding up the mutual information associated with the corresponding equivalent binary channels. This indicates the achievable rate if the underlying binary channels are *encoded* and *decoded* independent of each other. Referring to Fig. 1, we observe that there is a substantial gap of about 1.5dB in using Gray vs. Natural labeling. The curve labeled as "4-point constellation with equal probability of points" is the mutual information associated the original constellation when the constellation points are used with equal probability. This is indeed the sum of the achievable rates of the underlying binary channels if these are coded independently (using different code rates for the underlying binary channels), but the decoded sequentially where each decoder takes advantage of the decoded bit value produced by its previous decoders. The difference between this curve and the two curves labeled as "binary independent channels" accounts for the effect of the memory between underlying binary channels. The key point to improve the performance is to find a method to exploit this dependency towards increasing the achievable rate. The curve labeled as "4-point constellation with optimum probability of points" is the mutual information associated with the 4-point constellation maximized over the input probability assignment. The difference between this curve and those labeled as "4-point constellation with equal probability of points" accounts for the effect of shaping (using the constellation points with non-equal probability). Note that the shaping gain here is limited to about 0.5dB because the number of constellation points is small. We will later present numerical results for the achievable shaping gain in a 256 QAM constellation, showing a shaping gain more than one dB.

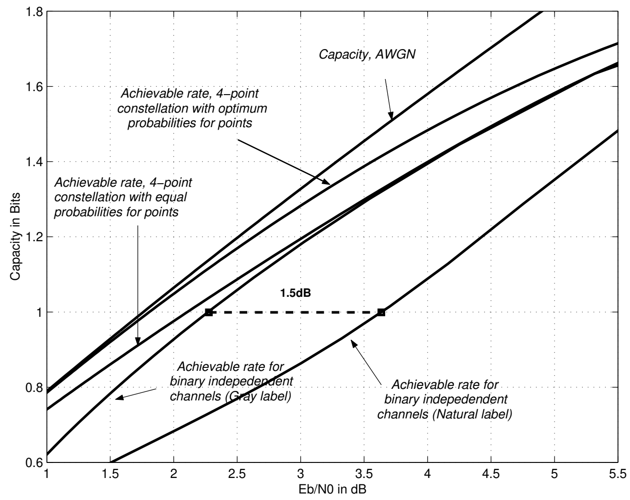

Fig. 1. Achievable rates in a one-D constellation with four points using different configurations, as well as the maximum achievable rate in an AWGN channel.

A method for exploiting the dependency between the binary channels is to incorporate the constellation as a component in the iterative loop where the probabilities calculated in a given step of the iterative process are passed as APP values to the constellation. A more practical method to exploit the dependency between the bits is to keep the dependent bits together in the process of encoding/decoding. This is achieved by interleaving several bits at a time [34]. The case of two bit interleaving is especially attractive because the number of branches starting from each state will be equal to four, resulting in two branches per bit [34][35]. As a result, the computational complexity (per bit) of the forward-backward algorithm on the resulting trellis will be the same as the standard case of single bit interleaving, while the effective length of the trellis reduces by a factor of two, reducing the memory required to store the state probabilities in the forward-backward algorithm by a factor of two. For larger values of m, the complexity of the resulting trellis for the symbol-based interleaving increases exponentially with m, making it infeasible. This makes the method unattractive for larger constellations. As an example, in Fig. 2, a 16-point constellation is decomposed into 4 cosets using the partition chin $Z/4Z$. These cosets are labeled by two bits using Gray labeling and are selected by the output bits of a symbol-based Turbo-code with $m = 2$. Such a coset decomposition of a large constellation results in the complexity of the symbol-based decoder to be independent of the number of constellation points (i.e., determined by the number of cosets). The decoding proceeds by: (i) detecting the sequence of cosets by decoding the Turbo-code (recovering part of the input bits), then (ii) detecting a point within the selected cosets, and finally, (iii) recovering the second sequence of bits using the inverse of the addressing scheme used for shaping. The probability values required to initialize the Turbo decoder are computed by adding the probability of points within each coset, and then the conventional iterative decoding procedure follows.

The achievable shaping gain for this constellation and the shaping partitions given in Fig. 2 can be computed using the technique given in [13]. The result for a constellation of dimen-

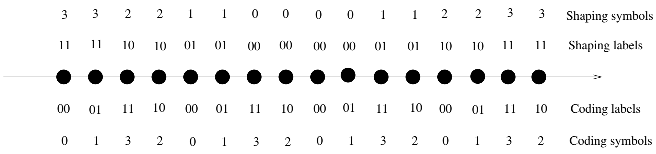

Fig. 2. A one-D constellation of 16 points divided into 4 cosets with related coding and shaping labels.

sion 32 as a function of Constellation Expansion Ratio (refer to [5] for definition) is shown in Fig. 3. The same graph also contains the shaping gain achievable in 32 dimensions if one does not group the one-D points into shaping partitions [12], as well as the shaping gain in an infinite dimensional space assuming continuous approximation.

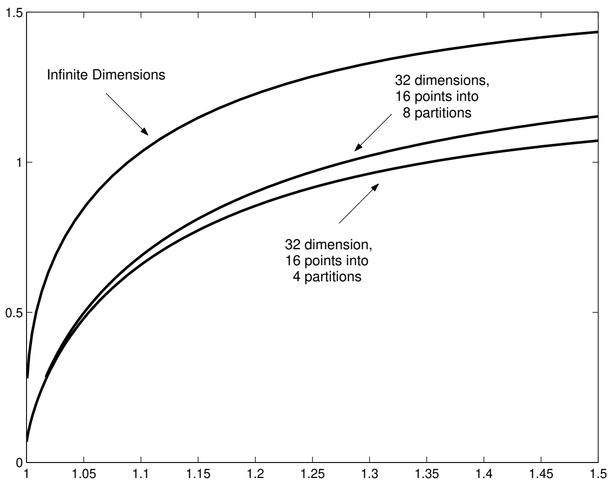

Fig. 3. Shaping gain in dB vs. CER for a 2-D sub-constellation with 256 points in a 32-D space.

We introduce an efficient addressing scheme for shaping in a 32-dimensional space with a CER equal to 1.35. We assume that addressing is achieved in a hierarchy of stages which each stage involves the Cartesian product of two (similar) lower dimensional sub-spaces. This results in 5 stages to reach to dimension 32 starting from 2-D sub-spaces. This hierarchy is shown in Fig. 4. To simplify addressing, the 2-D sub-constellations are divided into 16 energy shells of equal number of points with increasing average energy. This number of 2-D shells is enough to realize most of the achievable shaping gain for 2-D constellations with 256 (or more) points. For 2-D sub-constellations with smaller number of points, a lower number of 2-D shells (say 8) will be enough (this results in a noticeable reduction in the size of the look-up table). Note that in all our discussions, addressing refers to selecting a 2-D shell within each 2-D sub-constellation. Obviously, if each such 2-D shell contains 2^m signal pints, then one will need another m bits per 2-D to select the final point with the selected 2-D shell (in each sub-constellation). We do not discuss the effect of these extra bits because these are simply extracted from the input bit stream.

Assuming 16 shells in 2-D sub-spaces, we obtain $16 \times 16 = 256$ elements in the Cartesian product of the 2-D sub-constellations in 4-D sub-spaces. In general, we refer to the elements formed in the Cartesian product of the lower dimensional shells as *Clusters*. The 4-D clusters are ordered according to their average energy, and 16 subsequent clusters are merged into a 4-D shell. This results in 16 shells of equal cardinality in the 4-D sub-spaces. Again, the Cartesian product of the 4-D shells (resulting in 256 clusters in each 8-D sub-space) are ordered according to their average energy and subsequent clusters are merged into 8-D shells. To reduce the addressing complexity, the merging at this stage is achieved using non-equal number of clusters in subsequent 8-D shells. In specific, the number of clusters (in the order of increasing average energy) merged into subsequent 8-D shells are equal to $\{16, 16, 32, 32, 32, 32, 32, 64\}$, respectively. This results in 8 shells in 8-D sub-spaces. Following that, we obtain 64 clusters in the 16-D sub-spaces (which are not merged) and 4096 clusters in the 32-D space (which are not merged). Note that the cardinalities of the 16-D and 32-D clusters are not equal (these cardinalities are all an integer power of two). The final constellation is selected from the 32-D clusters of least energy such that the overall bit rate is 54. Note that without shaping, the bit rate would be $16 \times 4 = 64$. This means that the shaping redundancy is equal to 7 bits/32-D, resulting in,

$$CER = \frac{16}{2^{57/16}} \simeq 1.35$$

The addressing at the highest level of the hierarchy is achieved using a Huffman tree as shown in Fig. 5. The first step is decide if the selected 32-D cluster belongs to Set I, Set II or Set III. This is achieved by assuming that the 194 clusters are labeled by the binary number obtained by assigning zero to the left branch and one to the right branch at each node of the tree. Then, the label of each final node (corresponding to a cluster) is obtained by concatenating the binary labels of its branches (where the most significant bits correspond to values closer to the top of the tree). This will result in the labels of the final nodes to be ordered increasingly from left to right. Note that the label of the final nodes are composed of 7 bits (for Set I), 8 bits (for Set II), and 9 bits (for Set III). In this case, to select a 32-D cluster, we extract 7 bits from the input stream and compare its numerical value with the threshold T_1 which is the label of the last cluster in Set I. If the label is smaller or equal to T_1, then we are within Set I, and have another $57 - 7 = 50$ bits to proceed with the addressing within the selected cluster. Otherwise, we extract one more bit from the input stream, resulting in an 8 bit label. We compare the numerical value of the resulting 8-bit label with the threshold value T_2 which is the label of the last node in Set II. Again, if the label turns out to be smaller or equal, we proceed with the addressing within Set II using the remaining bits (in this case, 49 bits are left). If the label turns out to be larger than T_2, then we extract one more it from the input stream and use the resulting 9 bits to select an element within Set III (in this case, we are left with 48 bits to select an element within Set III).

After selecting one of the 194 clusters in 32-D (as explained above), to proceed with the addressing within the selected cluster, we assume that there exists a look-up table with 194 memory locations each of $4 \times 3 = 12$ bits, where the 3-bit address sections point to the 8-D shells building a given 32-D cluster (note that each 32-D cluster is simply the Cartesian product of four

8-D shells). This results in $1294 \times 12 = 2328$ bits of ROM.

To proceed with the addressing within 8-D shells, we note that the selected 8-D shell is composed of $16, 32$, or 64 of 8-D clusters (corresponding to 4, 5, 6 bits, respectively). In this case, we extract another $4, 3$ or 2 bits from the input (for each 8-D sub-space) to bring the total number of bits per 8-D to 8 bits. We use these 8-bit addresses to select an 8-D cluster in each 8-D sub-space. To do this, we assume that there exists a look-up table with 256 memory locations each of $4 \times 2 = 8$ bits, where each 4-bit section of these 8-bit addresses points to the 4-D shells construing a given 8-D cluster. These bits are used to select one 4-D shell in each 4-D sub-space.

To proceed with the addressing within 4-D shells, we assume that there exists a look up table of 256 memory location each of $4 \times 2 = 8$ bits, where each 4-bit section of these 8-bit addresses points to the 2-D shells construing a given 4-D cluster. These bits are used to select one 2-D shell in each 2-D sub-space. As mentioned before, there will be another group of input bits (depending on the number of signal points within each 2-D shell) which will be used to select the final point within each 2-D sub-constellation. Simulation results show that the proposed constellation offer a shaping gain of about 1 dB. The total memory requirement for the proposed addressing scheme is equal to,

$$M_{total} = 2328 + 2048 + 2048 = 6424 \text{ bits } \simeq 0.73\,K$$

Note that the clusters always happen in pair (of equal energy), say $A \times B$ and $B \times A$, unless the two constituent lower dimensional components are the same, say $A \times A$. This property can be used to reduce the size of the required memory by a factor of close to two at the price of a very small number of comparisons (comparing the labels to some anchor points corresponding to the label of the clusters with identical components).

Example: Assume that each 2D sub-constellation is composed of 256 points. The input is composed of a total of 57 shaping bits plus another 64 bits. We are only concerned with the operation of the selection of the 2-D shells using the 57 shaping bits. The selection of the final 2-D points within those 2-D shells is a trivial task. The same procedure will apply to any other scenario where the number of 2-D points is different from 256 while the number of 2-D shells is 16. Let us consider the following cases for the 57 shaping bits:

Case I: 0010010... The first 7 bits are 0010010 with the value of 18 that is less than $T_1 = 69$. In this case, the selection will be within Set I in Fig. 2. This value of 18 will point to a location in the corresponding look up table which contains 4 pointers each of 3 bits specifying the related 8-D clusters (there are 4 of these 8-D clusters). After knowing the 8-D clusters, we proceed with the addressing within each of those 8-D clusters by extracting an appropriate number of bits from the remaining input bits in a sequential manner and using those bits to select their related elements within the 4-D and subsequently 2-D sub-spaces by referring to the related look-up tables used for the addressing within 4-D and 2-D sub-spaces.

Case II: 1010010 0 100100 The first 7 bits are 1010010 which is equal to 82 and is greater than $T_1 = 69$. To proceed, we extract one more bit from the input which is equal to 0. Then, the first 8 bits, namely, 10100100 with the value of 164 is compared to $T_2 = 247$. As the label is smaller than T_2, then we are within the Set II. In this case, we are left with 49 shaping bits to proceed with the selection of the 8-D clusters.

Case III: 1111011 1 100100 The first 7 bits are 1111011 with the value 123 which is larger than 69 and the first 8 bits, namely 11110111 are of value 247, so we are within Set II and we use the first 8 bits for the addressing in Set II.

Case IV: 1111110 1 100100 The value of the first 7 bits is larger than T_1=69 and the value of the first 8 bits is larger than $T_2 = 247$, so we are with Set III and we extract the first 9 bits for the addressing with Set III

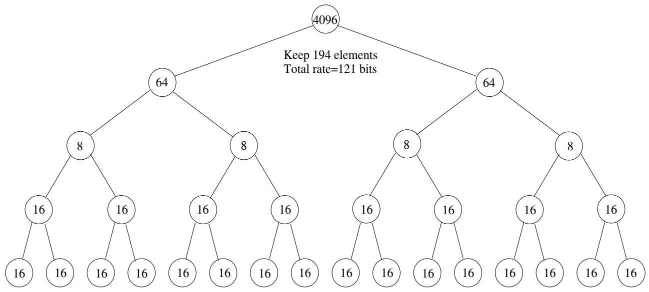

Fig. 4. Tree structure of the hierarchy used for addressing 32 dimensions.

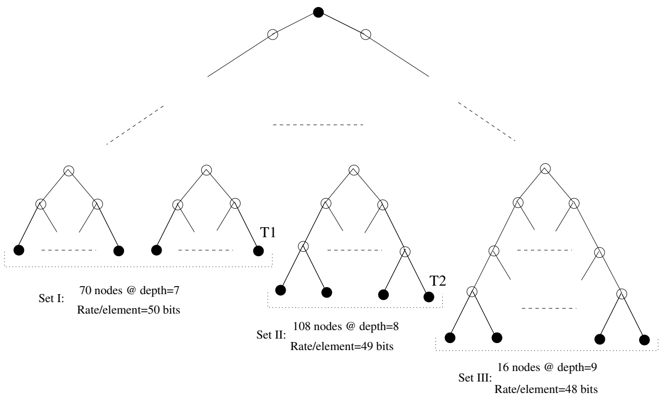

Fig. 5. Structure of the Huffman tree used for addressing.

Fig. 6 shows the BER performance of the proposed scheme for a spectral efficiency of 4,6 bits/sec/Hz. Block length is 16400 bits and the code has 4 memory elements, the interleaver is S-random with $S = 27$.

References

[1] J. L. Massey, "Coding and modulation in digital communication," Proc. 1974 *International Zürich Seminar on Digital Communication*, Zürich, Switzerland, pp. E2(1)-E2(4), Mar. 1974.

[2] G. Ungerboeck and I. Csajka, "On improving data-link performance by increasing channel alphabet and introducing sequence coding," *Proc. IEEE Int Symp. Information Theory (ISIT)* (Ronneby, Sweden), June 1976.

[3] H. Imai, and S. Hirakawa, "A new multilevel coding method using error-correcting codes," *IEEE Trans. Inform. Theory*, vol.IT-23, pp. 371-377, May 1977

[4] U. Wachsmann, R. F. H. Fischer and J. B. Huber, "Multilevel codes: theoretical concepts and practical design rules," *IEEE Transactions on Information Theory,* vol.45, no. 5, July 1999, pp. 1361-1391

[5] G. D. Forney, Jr. and L. F. Wei, "Multidimensional constellations–Part I: Introduction, figures of merit, and generalized cross constellations," *IEEE J. Select. Areas Commun.,* vol. SAC–7, pp. 877–892, August 1989.

[6] L. F. Wei, "Trellis coded modulation with multidimensional constellations," *IEEE Trans. Inform. Theory,* vol. IT–33, pp. 483–501, July 1987.

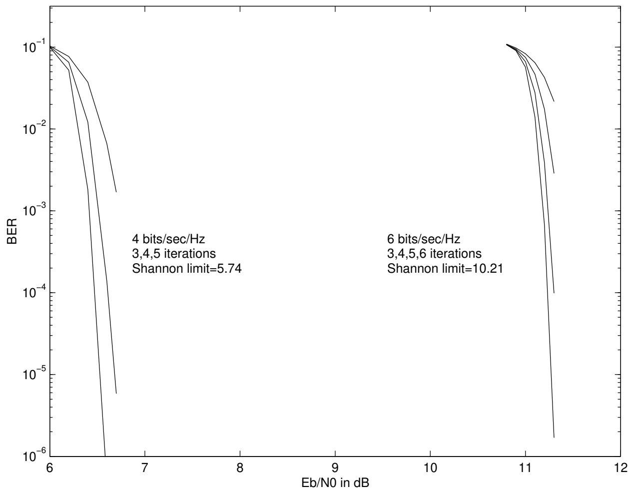

Fig. 6. BER performance of the proposed scheme for a spectral efficiency of 4,6 bits/sec/Hz. The best known result (equipped with shaping) for a scheme with a spectral efficiency of 4 bits/sec/Hz requires an E_b/N_0 of about 7.1 dB [39], and the best known result (without shaping) for a scheme with a spectral efficiency of 6 bits/sec/Hz requires an E_b/N_0 of about 14 dB [40].

[7] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Inform. Theory,* vol. IT-31, pp. 106–109, January 1985.

[8] G. D. Forney, Jr., "Multidimensional constellations—Part II: Voronoi constellations," *IEEE J. Select. Areas Commun.,* vol. SAC-7, pp. 941–958, August 1989.

[9] A. R. Calderbank and L. H. Ozarow, "Nonequiprobable signaling on the Gaussian channel," *IEEE Trans. Inform. Theory,* vol. IT-36, pp. 726–740, July 1990.

[10] G. R. Lang and F. M. Longstaff, "A leech lattice modem," *IEEE J. Select. Areas Commun.,* vol. SAC-7, pp. 968–973, August 1989

[11] P. Fortier, A. Ruiz and J. M. Cioffi, "Multidimensional Signal Sets through the Shell Construction for Parallel Channels", *IEEE Transactions on Communications,* vol. 40, no.3, pp. 500-512, March 1992

[12] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces-Part I: optimum shaping, shell mapping ," *IEEE Trans. Inform. Theory*, vol.IT-39, pp.1799-1808, Nov. 1993.

[13] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces-Part II: shell-addressed constellations," *IEEE Trans. Inform. Theory*, vol.IT-39, pp.1809-1819, Nov. 1993.

[14] F. R. Kschischang and S. Pasupathy, "Optimal shaping properties of the truncated polydisc," *IEEE Trans. Inform. Theory,* vol. IT-40, pp. 892–903, May 1994

[15] R. Laroia, N. Farvardin and S. A. Tretter "On optimal shaping of multidimensional constellations," *IEEE Trans. Inform. Theory,* vol. IT-40, pp. 1044–1056, July 1994

[16] G. D. Forney, "Trellis shaping," *IEEE Trans. Inform. Theory,* vol. IT–38, pp. 281–300, March 1992.

[17] F. R. Kschischang and S. Pasupathy, "Optimal nonuniform signaling for gaussian channels," *IEEE Trans. Inform. Theory,* vol. IT–39, pp. 892-903, May 1993

[18] A. K. Khandani and P. Kabal, "Shaping of multi-dimensional signal constellations using a lookup table," *IEEE Trans. Inform. Theory,* vol. IT–40, pp. 2058–2062, Nov. 94.

[19] A. K. Khandani and P. Kabal, "An efficient block-based addressing scheme for the nearly optimum shaping of multi-dimensional signal spaces," *IEEE Trans. Inform. Theory*, vol.IT-41, pp. 2026-2031, Nov. 1995.

[20] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes*, *IEEE Int. Conf. on Comm. 1993 (ICC'93)* (Geneva, Switzerland), pp. 1064—1070, May 1993.

[21] S. Le Goff, A. Glavieux, and C. Berrou, "Turbo-codes and high spectral efficiency modulation," in *Proc. IEEE Int. Conf. Commun.*, 1994, pp. 645-649

[22] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Bandwidth efficient parallel concatenated coding schemes," *Electron Lett*, vol.31, no.24, pp. 2067-2069, 1995

[23] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Parallel concatenated trellis coded modulation," Proc. *IEEE Int. Conf. Communications,* (Dallas, TX), pp. 974-978, June 1996

[24] U. Wachsmann and J. Huber, "Power and bandwidth efficient digital communication using Turbo codes in multilevel codes," *Europ Tran., Telecommun (ETT),* vol. 6, pp. 557-567, Sept.–0ct. 1995

[25] R. Pyndiah, A. Picart and A. Glavieux, "Performance of block turbo coded 16-QAM and 64-QAM modulations," Proc. *IEEE Global Telecommunications Conference, 1995* (GLOBECOM'95), vol. 2, pp. 1039-1043

[26] P. Robertson and T. Worz, "A novel bandwidth efficient coding scheme employing turbo codes," Proc. *IEEE Int. Conf. Communications,* vol.2, Dallas, TX, June 1996, pp.962-967

[27] P. Robertson and T. Worz, "Bandwidth-efficient turbo trellis-coded modulation using punctured component codes," *IEEE J. Select Areas Commun,* vol. 16, pp.206-218, Feb. 1998.

[28] L. Sylla and P. Fortier, "A study of Turbo codes for multilevel modulations in gaussian and mobile channels", Congres canadien de genie electrique et de genie infonratique, Waterloo, May 24-28, 1997

[29] H. Ogiwara and M. Yano "Improvement of Turbo Trellis-Coded Modulation System," *IEICE Trans. Fundamentals,* vol. E81-A, No.10, pp. 2040-2046, Oct. 1998

[30] T. M. Duman and M. Salehi, "Performance bounds for turbo-coded modulation systems," *IEEE Transactions on Communications,* vol.47 no.4, April 1999, pp. 511-521

[31] C. Fragouli and R. D. Wesel, "Turbo-encoder design for symbol-interleaved parallel concatenated trellis-coded modulation," *IEEE Transactions on Communications,* vol. 49, no. 3, March 2001, pp. 425-435

[32] C. Fragouli, R. D. Wesel, D. Sommer and G.P. Fettweis, "Turbo codes with non-uniform constellations," *IEEE International Conference on Communications (ICC 2001),* pp.70—73 vol.1

[33] Mark Bingeman and Amir K. Khandani, *Design, Performance Analysis and Decoding of Combined Turbocode and Modulation*, Proceedings of 19th Biennial Symposium on Communications, Queens University, Kingston, Ontario, Canada, pp. 36—40, June 1998.

[34] M. Bingeman and A. K. Khandani, "Symbol-based Turbo-codes," *IEEE Communication Letters*, Vol. 3, No. 10, pp. 285—287, Oct. 1999.

[35] M. Bingeman, A. K. Khandani and F. Khaleghi, "Symbol-based Turbo-codes", United States Patent #6,298,463

[36] P. Sauvé and F. R. Kschischang, *Decoding Turbocodes by Multibit Probability Propagation, Proceedings of 19th Biennial Symposium on Communications,* Queens University, Kingston, Ontario, Canada, pp. 89—93, June 1998,

[37] C. Berrou and M. Jezequel, "Non binary convolutional codes for turbo coding", *Elect. Letters,* vol.35, no.1, pp. 39-40, Jan. 1999

[38] C. Berrou, M. Jezequel, C. Douillard and S. Kerouedan, "The Advantages of Non-Binary Turbo Codes," Proc. *2001 IEEE Information Theory Workshop,* Cairns, Australia 2-7 Sept. 2001

[39] S. Benedetto, D. Divsalar, G. Montorsi and F. Pollara, "Parallel concatenated trellis coded modulation," IEEE International Conference on Communications, 1996 (ICC96), Page(s): 974 -978 vol.2

[40] H. R. Sadjadpour, "Application of turbo codes for discrete multi-tone modulation schemes," *IEEE International Conference on Communications,* 1999 (ICC99) Page(s): 1022 -1027 vol.2