Polynomial Calculation of Quasi-Gray code Labelings for Irregular Signal Constellations

Jaimie Jellema
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
j.m.jellema@student.utwente.nl

ABSTRACT

The commonly used quadrature amplitude modulation (QAM) constellation has a shaping loss of about 1.53dB compared to the Additive White Gaussian Noise (AWGN) Shannon capacity. Usage of non-square constellations can improve this. However, non-square constellations exhibit Gray-code loss due to the impossibility of perfect Gray-code labeling, on constellations which do not have an orthogonal basis [32]. Labelings of constellations can be improved using non-linear constraint solvers, but this exhibits large computational costs for larger constellations.

The Binary Switching Algorithm (BSA), which can produce very good labelings, is non deterministic. BSA is an algorithm based on hill climbing from many starting points to produce the assumed global optimum. This paper presents a rule-based algorithm based on the KD-tree algorithm. The proposed algorithm produces labelings which offer a good approximation of Gray-code labeling. Gray-code based labelings can offer undominated performance in a variety of situations [36][8]. The algorithm as proposed can be seen as a method to extend the Symmetric Ultra-composite (SU) method as proposed by Wesel et al [36] to arbitrary signal constellations. The proposed algorithm has a polynomial time complexity, namely $\mathcal{O}(n \log^2(n))$. Enabling the creation of good labelings at very low computational cost. This method is applied to and optimized for the Golden Angle Modulation (GAM) constellation [26]. This method produced better performance in uncoded situations than the labeling as created by Larsson. The method reduced the needed signal-to-noise ratio by 0.1 to 0.2dB for a Bit Error Rate (BER) of 10^{-3} to 10^{-6}. For very low Signal-to-noise Ratio (SNR), the method also obtained better BER performance across the DVB-S2 constellation with Low-Density parity check codes error correction.

Keywords

Golden Angle Modulation, constellation mapping, constellation labeling, mutual information, KD-tree, BICM, BICM-ID, DVB-S2

1. INTRODUCTION

All research starts by gathering information. For many space missions, the problem is not gathering data, the problem is getting the data back to Earth so research can be performed on it.

For many space probes, the transmitter antenna is more resource constrained then the receiver antenna. This can be seen by for example the Voyager-1 probe only having a 3.7m dish [23], while the receivers consist of 70m antenna with helium cooling [27]. This communication is traditionally limited by the transmission power of the satellite and not the available bandwidth [24]. Hence in order to get the highest amount of data from our satellites, we need to search for the most power efficient way to transmit data back to earth.

1.1 Modulation

There are two basic properties of a signal that can be changed in order to send information: the amplitude, and the phase of the signal. Modifying the properties to send information is called modulation. In order to transmit data, the phase or amplitude can be set to a certain value. The range of amplitudes and phases that can be transmitted can be put in a 2D-chart, this chart is called the constellation map. A particular arrangement points in the map is called a constellation. In this constellation map, the distance from the center is a measure of signal power, and the angle is the phase with which the signal is transmitted. In a constellation map the horizontal axis is commonly called I, and the vertical axis is called Q. A very common constellation is the Quadrature Amplitude Modulation (QAM) [18], which can be seen in Figure 1. QAM is used in, among others, ADSL, WiFi and the digital television standard (DVB-T).

However, QAM is optimized for easy modulation of signals and high throughput, not power efficiency. There has been significant research into creating more power efficient constellations, which usually means a more complex decoding process. One of the resulting modulations is Golden Angle Modulation (GAM) which is a modulation constellation proposed by Peter Larsson et al. in [26]. It can be seen that GAM as seen in figure 2 has a more circular shape then QAM, seen Figure 1. This circular shape helps to improve power efficiency. Which is useful since for satellites the peak transmission power, rather then the average power is restricted [10]. The improvement in performance obtained by using a more circular constellation, instead of a square constellation is called shaping gain.

1.2 Constellation Labeling

In order to create a working communication system, conversion from data to constellation points has to be chosen. The process of assigning indexes to a constellation is called labeling. The performance of the labeling in an

31th Twente Student Conference on IT July 5th, 2019, Enschede, The Netherlands.

uncoded Additive White Gaussian Noise (AWGN) channel. Is influenced by the labeling. Since when a symbol is incorrectly received, it most likely will be received as one of the symbols very close to the transmitted symbols. If the labelings have a bit pattern which is similar, then receiving the wrong symbol, will lead to less bit changes. The amount of bit changes needed to go from one label to another is called the Hamming distance. The straight forward labeling as proposed by Larsson suffers from the problem that the bit patterns of constellation points that are close together are not at all similar, having a large Hamming distance. This can be seen in Figure 2 where the points with value 5 and 10 are relatively close together, even though they have the maximum possible Hamming distance of 4. Whereas for QAM, as shown in Figure 1, the values of points that are close together have a small Hamming distance. Due to both the spiral nature and being based on the golden ratio, the GAM constellation is highly irregular. Due to being so irregular, designing bit patterns in such a way that the Hamming distance is low, is non-trivial.

The problem of a large Hamming distance between adjacent symbols is that a small amount of noise can result in another symbol being received, which has a completely different bit pattern. If the symbols have closer bit patterns and hence are so called quasi-Gray coded. It is easier to compensate using error correction [3]. This means a lower degree of error correction can be used, which means we can send more data over our communication channel for the same amount of power. A class of constellation labelings which is especially capable of reducing Hamming distance between constellation nodes is Gray-code and quasi-Gray-code based labelings.

The ideal labeling in a coded channel is dependant on the amount of *a priori* channel information that is known to the receiver. Different labelings can be optimized for various amount of *a priori* information. Labeling research is especially interesting and important because it is one of the few areas of telecommunication where the performance can be improved by pre-calculation of a lookup table, and does not require a more complex or powerful decoder architecture. This means any gain which can be obtained by means of improving the labeling of the constellation, is essentially for free.

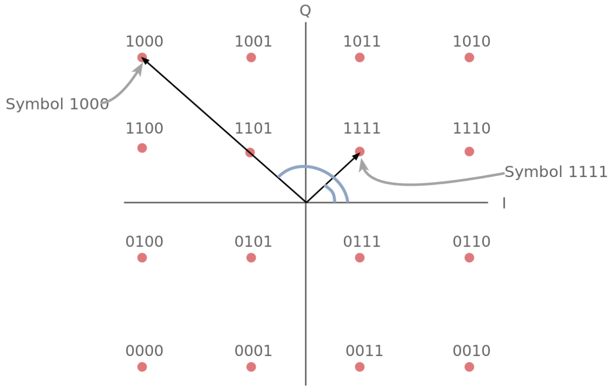

Figure 1. 16-point QAM Gray-code labeling

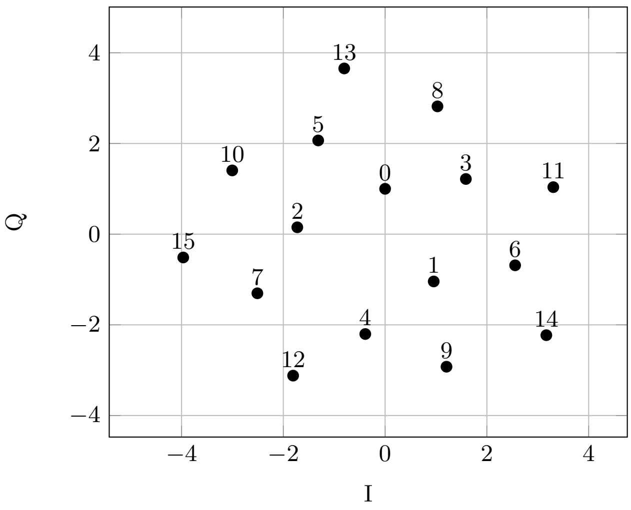

Figure 2. 16-point GAM constellation Larsson labeling

Dec	Binary	Gray
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Table 1. Comparison of Binary and Gray-code

1.3 Binary Reflected Gray Code

The binary number systems has a disadvantage for numbering of constellation labelings in that when a transition is made from BLARGH to BLARGG a large amount of bit changes is needed. Binary Reflected Gray Code is a re-ording of the binary number systems, which reduces the Hamming distance between consecutive elements to one. The binary reflected Gray code list for n bits can be generated recursively from the list for $n-1$ bits by reflecting the list (i.e. listing the entries in reverse order), prefixing the entries in the original list with a binary 0, prefixing the entries in the reflected list with a binary 1, and then concatenating the original list with the reversed list [12]. In Table 1 the first numbers for BRGC are given.

1.4 Binary Switching Algorithm

Currently the best known algorithms for generating labelings are modified versions of the Binary Switching Algorithm (BSA). The BSA is a greedy algorithm which works from a random starting state. From this starting position it calculates which constellation label is making the greatest contribution to the Bit Error Rate, and swaps this label with the label which would give the greatest improvement to the mapping. Then it continuous searching for the next worst label, switching it for the point which gives biggest improvement, continuing in this fashion until no more improvements are possible. This means a local optimum is reached, and if enough starting points are tried, the assumed global optimum [30]. The BSA algorithm requires a cost function which can be used as an indicator for the performance of the labeling in the channel it is used. The BSA algorithm will try to minimize the cost function. It was shown by Xingyu Xiang et al. in [37] that a lower cost function does not necessarily constituted to better performance. Therefor the labelings which produce a low cost function are run through a simulation in order to get the best performing labeling.

This strategy starts breaking done for bigger constellations were it becomes computationally infeasible to test many different starting positions, and evaluate the cost function. Using rule based techniques Navazi et al. were able to obtain 4.5dB gain compared to an speed optimized version of the BSA algorithm [29]. They were unable to execute a single round of 4 dimensional 1024-QAM, due to computational complexity involved in the calculation.

1.5 Symmetric-Ultracomposite

A method for labeling of constellation points is the Symmetric-Ultracomposite (SU) labeling, as introduced in [36]. The SU labeling produces constellation labelings which provide undominated performance across a wide range of signal-to-noise ratios. The SU labeling as proposed can only be applied to square QAM constellations, allowing the creation of both Gray and Set-Partioning based labeling. In an effort to extend the SU labeling to no square constellations, the quasi-SU labeling is also introduced. These quasi-SU labelings tries to closely mimic Gray-code structures. This quasi-SU as proposed only works for cross-QAM. It is recommended by Wesel et al, that future labeling research should focus on labeling of other constellations aswell.

1.6 Calculation of uncoded BER of labeling

In order to test a labeling method a measure of performance is needed. The Bit Error Rate is on such measure of performance. The Bit Error Rate for a particular constellation, is dependant on the labeling and the Signal-to-Noise Ratio (SNR). In order to evaluate the effect of labeling, the labeling is tested across a range of SNR values. Then a graph can be created of the Bit Error Rate over the SNR value. The best labeling is the labeling which requires the lowest SNR for a particular BER. There are two main ways to test a range of SNR values for a labeling. One method is stochastic, and one method is exact. The stochastic method simulates a high amount of bits across the channel, and records the error rate. This provides a good estimate of the BER, if the amount of bits is high enough. The exact method calculates the probabilities that a symbol is incorrectly received. Using these probabilities the BER can be calculated. In order to allow testing of as many constellation labelings as possible in a given time frame, GPU-acceleration for calculating BER of a labeling has to be considered.

1.7 Problem Statement

The goal of this research is to create a method of labeling the irregular constellation points in a way that obtains an as low as possible Bit Error Rate between the sender and receiver of the communication system. It should achieve these results across a given signal-to-noise ratio, of interest for telecommunication.

Due to GAM irregular nature as explained in Section 1.2, it provides a good test case for experimentation with new labeling strategies.

GPU-accelerated calculation of the performance of the various labelings could be a useful tool in this research. Since GPU-acceleration could possibly offer fast and accurate results for the performance of a particular labeling.

1.7.1 Research Questions

1. To what extent can the bit error rate for irregular constellations based communication system be improved using a different constellation labeling
2. To what extent can GPU acceleration help in evaluating the performance of a constellation mapping
3. How can our proposed solution be integrated in transceiver hardware

2. RELATED WORK

The amount of research on the GAM constellation is fairly limited since it's a relatively new modulation technique. GAM can be seen as a result of a continuous search for more power efficient transmission of information. Larsson experimented with varying the radial distances of the constellation points in GAM. Furthermore, genetic algorithms have also been used to optimize the positioning of constellation points in [4].

Other research that were trying to obtain power efficiency, achieved this by trying to mitigating shaping loss. Hexagonal structures [19], instead of square structures, allow for a denser packing of constellation points which helps improve power efficiency. Another way to improve efficiency is by reducing the Gray-code loss as a result of non-square shaping of the constellation [32]. There have also been ways to extend Gray code mapping to non QAM constellations The trade-offs in constellation shaping are extensively discussed in [16].

As far as labeling research is concerned, Xiang et al. in [37] researched how to efficiently assign labels to a 32 point Amplitude Phase Shift Keying (APSK) constellation map. This obtained a performance gain of about 0.13dB compared to the standardized DVB-S2 mapping. F. Kayhan et al. in [25] also attempted this using a different technique. They constructed a labeling of the constellation were all the rings are Gray-coded, obtaining a 0.15dB increase in performance.

There has also been research on labeling by means of searching for local optimum from a random initial labeling, such as BSA [30]. However these methods provide no insight into efficient techniques for constructing such a mapping, nor will it elegantly extend to higher order system. Higher constellation orders are needed for higher spectral efficiency. It can be shown that the labeling problem is equivalent to the Quadratic Assignment Problem (QAP), and for a small number of constellation mappings, optimal solutions are known [40]. For standard PAM, PSK, square-QAM, and cross constellations, a framework for labeling was created in [36].

With regards to the theoretical framework regarding the optimality of the Gray-code based techniques, Alvardo et al. said the following in [3]: *"It is also worth mentioning that for BICM it was conjectured in [6] that a Gray mapping maximizes the BICM capacity given by [38] . . . This conjecture has recently been disproved in general in [33], however, it seems that the optimality of Gray mappings, and particularly of the binary reflected Gray code (BRGC), holds for most of the relevant cases. Additionally, it is worth mentioning that Gray mappings offer small improvements through the iterations in BICM-ID."* (paraphrased) So even though Gray coding constellation labeling, may not always be optimal. It still is a powerful tool to have, considering in most relevant cases it is useful.

Instead of minimizing the the Hamming distance with nearby constellation labelings, an other possible goal is to maximize the squared Euclidean distance of constellation points with a Hamming distance of one to the current label. An other technique is called set-partitioning. Set-partitioning, partitions the space and with each subset the minimum Euclidean distance between signal points is maximal for the adopted subset size and is uniform across the subsets [7]. Although these techniques can offer better performance, they are outside the scope of this paper.

3. METHODOLOGY

In order to answer the research questions properly, the methodology is split up into two parts. The first part is creating methods which generate efficient constellation labels. The second part is efficiently evaluating the performance of those labeling methods under varying signal-to-noise ratios. The evaluation has to be efficient, and have a low time complexity, in order to finish the research in the given time frame.

3.1 Creation of Constellation labelings

In order to create efficient constellation maps, inspiration is drawn from the Gray-coded QAM constellation, which is shown in Figure 1. This is because Binary reflected Gray-coding is shown to be the optimal way to assign labels to these traditional QAM constellations [1]. In the rest of this paper, BRGC should be assumed when Gray-code is used. In order to see how Gray-coding might be used to label constellation points, a more abstract, general method of Gray labeling has to be defined first. For a constellation testbed, the GAM constellation was chosen, since it is a highly irregular constellation. Meaning a method capable of labeling GAM with good performance, can be probably be adapted to other constellations with similar performance improvements.

Note that, in order to generalize our problem, the bits in QAM constellation can be seen as a form of recursive binary space subdivision. See Figure 1, that the leftmost bit divides the constellation map in an upper and lower half, and the third bit defines a left and right half of constellation map. This grouping determines which labels need a low Hamming distance between them. Our method for the creation of the labelings for a constellation is a two-step process, step one is creating a way to group all constellation points which should have similar labelings. The second part is to, based on these groupings, label the constellation points.

3.2 Splitting Strategies

For the first part, the of grouping the constellation points, a space partitioning algorithm can be used. An example of space partitioning algorithm is the K-dimensional tree algorithm (KD-tree). The KD-tree algorithm especially is useful because it creates a binary tree. In our method, the constellation points are the leaves of the binary tree created by the KD-tree algorithm. In order to allow labeling later, a binary tree is needed, which is why the KD-tree algorithm is used instead of other space portioning algorithms. In order to see how such a tree can be created from a 2D space containing 2^N constellation points, where N is the number of input bits. The space is split into 2 partitions, with each partition containing 2^{N-1} constellation points. This splitting is done by inserting a line or line segment. This splitting is done recursively, each time halving the remaining constellation points inside a subspace. This is continued until the smallest subdivision only contains one constellation point. The manner in which the space is split is an important parameter in the performance of the final created labeling. Since the splitting strategy determines how the grouping of bits is done. Different grouping strategies of the labels are explored in the following sections.

These methods can be compared to both, each other and the assignment as proposed by Larsson. We can also calculate the effect of swapping any two points in the constellation mapping created by our proposed method, if the swapping of any two points results in better results. In that case, our method is not locally optimal, and hence also not globally optimal. This was not done in our research.

3.2.1 *KD-tree axis-aligned splitting planes*

This method creates splitting planes which are aligned with the I and Q axis of the constellation. The choice of which plane to start splitting on is arbitrary due to order of the bits in the labeling not mattering, producing labels which are in the same equivalence class.

In our algorithm the first splitting plane is parallel with the Q axis. After that, on each half of the constellation mapping induced by this splitting, a new splitting plane is inserted, only this time the splitting plane is aligned to the I axis, this process of alternating between splitting on Q and I continues until each subdivision contains one constellation point. The result of this labeling and the resulting splitting planes is shown in Figure 3 for an arbitrary constellation which needs to be labeled. The order of insertion of splitting planes is blue, light blue, green, yellow. Each new splitting line divides the remaining space in the subdivision in two. So the light blue division lines partitions the two subspaces created by the blue division line.

The motivation for this method is that it is easy to implement and that it produces clustering-like behaviour of similar labels.

This labeling can create pairs of points with higher Hamming distance than might be expected. Due to the recursive nature of the strategy, splitting planes of sibling nodes can be rather far apart. This can bee seen in Figure 3, where point *0010* is rather close to *0001* even though the have a Hamming distance of two. This edge effect is rather minor, and unavoidable given the constellation and mapping technique.

A further test of labeling is to terminate the algorithm earlier then full depth execution. This means that there is more then 1 node in each subsection. Here, the algorithm is tested to a depth of 4, which for a 256 point constellation means that there are 16 points in each subsection. The first 4 layers of the tree determine the first 4 bits of the labeling pattern, and the remaining 4 bits are based on the sorting them remaining labels and apply Gray-code binary to the ordered list per subdivision of the constellation space.

3.2.2 *KD-tree polar coordinates*

The motivation behind splitting based on polar coordinates is that the GAM constellation is inherently circular in nature, and hence, using polar coordinates appears to be a natural fit. Here, the first splitting plane can more accurately be described as a circle centered at the origin of the constellation diagram. The second splitting plane is based on the angle each point makes with the origin, where the splitting plane is put at the angle that divides the constellation points within the circle sector in half. Then the resulting circle segments are again split based on the radius, followed by a split based on the angle, this process again continues recursively until each circle segment contains exactly one constellation point. This is illustrated in Figure 4. Same color schema is used as in Figure 3 described in Section 3.2.1.

3.2.3 *KD-tree cross*

A solution to these edge effects of the splitting as mentioned in the Section 3.2.1. Is to also have splitting planes

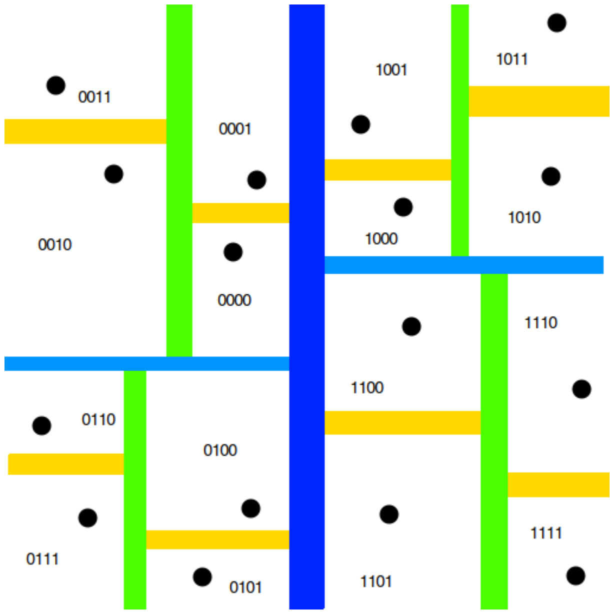

Figure 3. KD-tree axis-aligned splitting and numbering

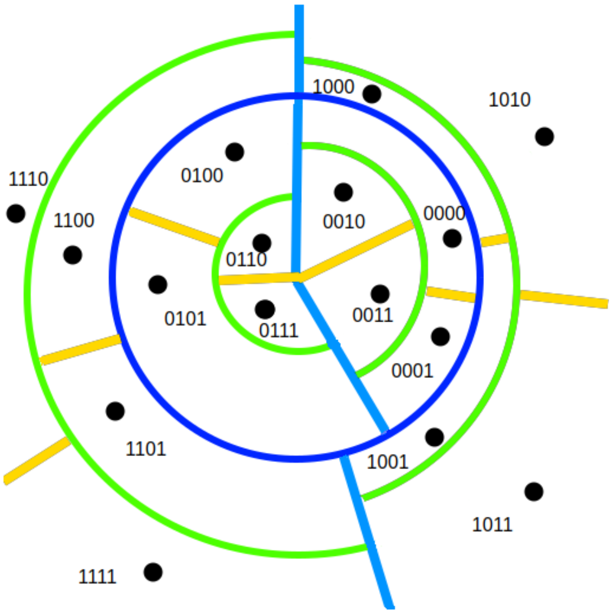

Figure 4. KD-tree polar splitting and numbering

which are not aligned to the axis of the coordinate system. This method consists of creating splitting planes at 45 degrees with respect to the coordinate system. The two splitting planes which are created this way are then alternated with the axis-aligned variant see Figure 5 for illustration of exact splitting. Same color schema is used as in Figure 3 described in Section 3.2.1.

3.2.4 Creation of binary tree based labels

The starting point for deciding how the assignment of labels should be done is that, when this labeling is applied to the QAM constellation, it should Gray-code the QAM constellation.

The labeling strategy should minimize the Hamming distance between constellation points. The Gray-code labeling of the QAM constellation is an example of a labeling which minimizes the Hamming distance with neighbouring points. Therefore we try to design the labeling strategy in such a way that when try strategy is applied to the QAM constellation, it produces the Gray-code labeling.

Now the Binary Reflected Gray Code is a key point here. In the same way that BRGC is created by starting from just two labels, which are mirrored and prepended with either 0 or 1. The same the binary tree can be mirrored and the branches of the tree given a 0 or 1 as part of their label. Hence in order to Gray-code label the constellation points, The tree created from the KD-tree algorithm has to flip the sub-tree of every right child node. This creates the same binary reflection that is used to create Binary Reflected Gray Code as described in Section 1.3. This flipping of the tree is equivalent to when traversing a binary tree in infix order, each time when entering a right child of the tree flipping its two child nodes.

After this flipping has been applied to the tree, the nodes are labeled by traversing the tree from root node to all the leaf nodes again, prepending a 0 each time a left child node is taken, and prepending a 1 each time a right child node is taken. The code that was used in this research to obtain these results are listed in Appendix A.

Figure 6 shows the relation between Binary Reflected Gray Code, and a binary tree which labels the constellation points. Every time the traversing follows a red branch, the children of that sub tree are swapped.

3.3 Transforming Gray-Code to Anti-Gray-Code

The Gray-code constellation is an interesting constellation for reducing BER in AWGN channels, however for BICM-ID channels, Set Partitioning based methods tend to perform better [28] for higher SNR values.

The creation of a Gray-coded constellation is a powerful tool in its own right. The anti-Gray code labeling is a test to see, if a simple change to labeling algorithm allows for labeling strategies which have different properties. The properties of the other labelings can be beneficial in other communication channels.

Code diversity is obtained by having a high Hamming distance between nearby constellation points in the rest of this section, a mapping is explained from low Hamming distance quasi-Gray-code labeling, to a high Hamming distance quasi-Anti-Gray code constellation. This labeling should offer better performance in BICM-ID Rayleigh fading channels [5][20].

In case of Gray code, and the standard QAM constellation, we can obtain a set of ideal labelings which have a Hamming distance of 1 with all the orthogonal neighbouring nodes. For Anti-Gray code constellation with size N, the upper bound on the average Gray-Code distance between orthogonal neighbours can be derived. This result is presented Expression 1

$$\frac{\frac{N}{2} \cdot \log_2(N) + \frac{N}{2} \cdot (\log_2(N) - 1)}{4(N-2)^2 + 12(N-2) + 8} \tag{1}$$

This expression can be derived as follows: Since in N constellation points which each have code length $log_2(N)$, there can be at most $N/2$ pairs of codes with exact opposite bit patterns. Hence the other nodes must have a distance lower then $log_2(N)$. The total number of orthogonal neighbours is the sum of: $(N-2)^2$ inner nodes with 4 neighbours, and $4 \cdot (N-2)$ edge nodes with 3 neighbours,

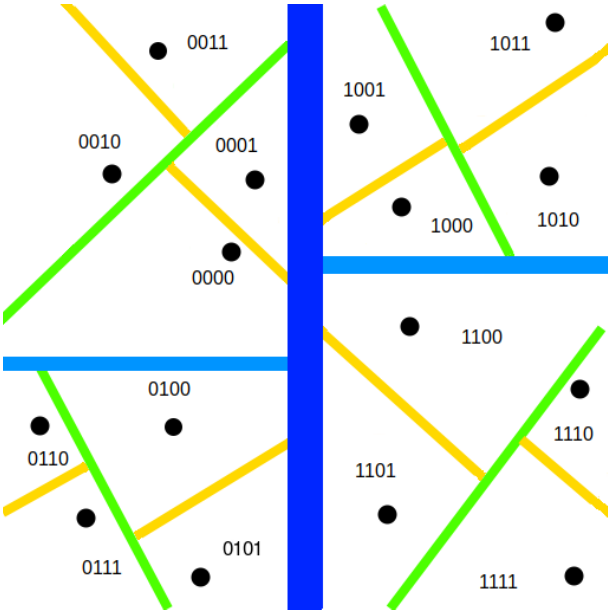

Figure 5. KD-tree cross splitting and numbering

and 4 corner nodes with 2 neighbours.

The construction of Anti-Gray-code of size 2^N can be done by starting with a BRGC sequence of size 2^{N-1}. Half the elements can be created by prepended with a 0. The remaining half can be created by inverting all elements from the original sequence and intersperse them with the elements previously obtained. This anti-Gray-code sequence will have a Hamming distance alternating between N and $N-1$ for consecutive elements.

Now an anti-Gray-coded constellation can be created by creating the Gray code sequence of length 2^M and the anti-Gray code sequence that has length 2^M. Subsequently a mapping can be created from Gray-code to anti-Gray-code, by using the index of an element in the Gray-code sequence as index in the anti-Gray code sequence. This bijective mapping can then be applied to a constellation created using the proposed method from Section 3.2. Applying the mapping will result in a constellation which maximises the Hamming distance between adjacent nodes, as consecutive elements which had low Hamming distance will be mapped to pairs of elements with a high Hamming distance.

3.4 Evaluating Constellation Labeling Performance

Two different methods for evaluating the performance of the constellation labeling will be addressed. One stochastic method, and one hardware accelerated based approach. The Monte-Carlo simulations evaluates the Bit Error Rate after a certain amount of symbols have been transmitted through the communication channel, which have been transmitted using our labeling. This means writing a software simulator which can emulate the communication channel, under the assumption of Additive White Gaussian Noise (AWGN). Each labeling can then be tested at various signal-to-noise ratios and see how the Bit Errors Rate is affected by the proposed labeling.

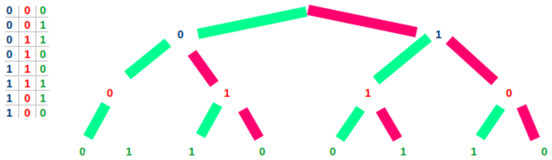

Figure 6. Illustration relation Gray and Tree labeling

This simulator can be used to verify the working of a hardware accelerated approach, in this case by using GPUs and Compute Unified Device Architecture (CUDA). This hardware-based approach will calculate the probabilities that constellation points will be incorrectly received at the receiver, for a given signal-to-noise ratio, which can then be used to calculate the resulting bit errors and hence the Bit Error Rate.

3.4.1 *Stochastic Simulator*

For the stochastic simulator a C++ based program was written. This simulator is capable of simulating the transmission over an Additive White Gaussian Noise (AWGN) channel. It can test the Bit Error Rates under a diverse amount of signal labelings. It makes use of a Mersenne Twister pseudo-random number generator (PRNG), which is seeded by the random number pool of the Operating System. This PRNG is used to both generate a random symbol to transmit, and add normal distributed noise to the received signal. This received signal is then mapped to the symbol with the highest likelihood, which for this equidistributed case means the nearest symbol in Euler distance.

The transmitted and received signal can then be compared, and the Hamming distance between the two symbols can be added to a running total of the total number of bit errors.

3.4.2 *Hardware acceleration*

In order to achieve significant speed-up in the calculation of the performance of the various constellation labelings, a graphics card can be used. There exists an obvious trade-off between precision and the computational power spent on calculating the Bit Error Rate.

Calculating the Bit Error Rate of a specific constellation and labeling can be done in a two-step process. Step one is an accurate estimation of the probabilities that symbol X is received when symbol Y is transmitted for all possible pairs of sent and received symbols.

Step two of this process is to calculate the contribution to the total bit error rate for all possible combinations of sent and received symbols. This can be obtained by calculating the Hamming distance between the send and received symbol, which is multiplied by the probability of this pair occurring. The relevant equation is given in Equation 2.

$$\begin{aligned} BER = \sum \Big(& Hamming(send, received) \\ & \times \\ & prob(send, received) \Big) \end{aligned} \tag{2}$$

The Hamming distance between two labels can be efficiently computed by taking the exclusive or operator on the binary representation of the labels, followed by counting the numbers of bits that are set. Which can be efficiently calculated by making use of the *popcount* intrinsic instruction of CUDA.

In order to obtain an accurate estimation of the transition probabilities, two possibilities can be considered. One method uses the exact approach as detailed in [34], however this requires quite a bit of effort to implement. Since it requires calculating the cumulative distribution function of a bivariate Gaussian random variable in decision regions. It was deemed that the risk of programming errors invalidating the results of the method was not worth the effort, given the time constraints. Instead the second option was chosen, which was to use a Monte-Carlo based method. For each possible constellation point, it was transmitted 2^{20} times and the resulting received symbol after applying noise was recorded. This was repeated for various values of SNR, this gives a good approximation of the transition probabilities for various values of SNR.

3.4.3 Coded performance

Most telecommunication systems that are currently in use, make use of Bit Interleaved Coded Modulation (BICM), which means a concatenated architecture of an outer error correcting code, a bit-interleaver, and an inner correcting code [3]. Usually the inner code is a soft decision based code such as convolutional, Turbo, or Low-Density Parity Check (LDPC) codes. The outer code is then usually a block code such as Reed-Solomon or BCH.

The reason for this architecture is that it allows to correct large burst errors for relatively modest sizes of the individual correcting codes. There is a subset of error correcting codes, based on an iterated sparse graph architecture, that exhibit a phenomenon known as 'error floor'. Turbo codes and LDPC are examples of such codes. If the BER is plotted over the SNR, then as the SNR increases, the BER becomes lower. For higher SNR values the BER drops more and more rapidly. For codes that have an error floor, they will reach a threshold SNR value, after which the BER does not drop as quickly as before. The concatenated architecture means that if the inner code exhibits error floor, the outer code is able to correct it.

In order to test the underlying assumption of the labeling technique, a better AWGN performance results in better coded performance, The following simulation setup was used, based on the DVB-S2 system. The specific settings were:

- 32 point constellation
- Coding rate 4/5
- Ring ratios of (2.87, 4.87)
- 30000 frames consisting of 648000 bits each
- BER is taken after LDPC decoding

The BER is measured after LDPC decoding and not after BCH decoding. This should not make a major difference since the BCH outer code is a hard-decision block code, the LDPC error rate is a decent indicator of the post-decoding error rate [39]. The testing of coded performance was stochastic in nature, and in order to accurately detect Bit Error Rate as low as 10^{-6}, a high amount of computational resources was required. This could have been mitigated by making use of importance sampling techniques, as illustrated in [14][11][21]. Due to time constraints, it was not possible to implement any of the techniques presented in those papers.

4. RESULTS

In this section, the uncoded AWGN channel and labeling will be covered first. Subsequently, the performance of the labeling strategies when using bit-interleaved coded modulation error correction will be evaluated. Afterwards, the effectiveness of the GPU accelerated approach in calculating the BER of the constellation labelings will be covered. Finally, the necessary hardware changes needed to integrate the proposed method in current hardware will be discussed.

4.1 Comparing various labeling strategies

From the results in Figure 7, we conclude that the new labeling method was able to improve the BER for the complete range of SNR covered. The jittering in the low BER range is probably due to limitations of the Monte-Carlo method in simulation of rare events. There are not enough samples of these rare events to provide accurate estimation of the exact probability. Due to this reason the graph was cut short from the original 35dB to 20dB, since those SNR ranges the rare events were so unlikely, that they were not at all detected. Figure 8 shows that the new labeling was able to reduce the SNR required to obtain a BER in the range of 10^{-3} to 10^{-6}. The difference measured is 0.1 to 0.2dB lower SNR requirement. Furthermore it can be seen that the anti-Gray code generation requires higher signal-to-noise ratio than the Larson labeling, this suggests that our remapping method was effective at reaching a higher Hamming distance between nearby constellation points.

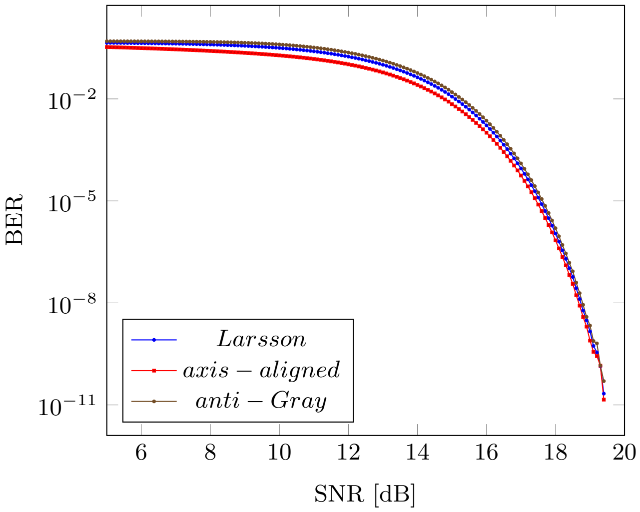

Figure 7. BER versus SNR of various labelings, constellation size 256

Figure 8 shows that all of the labelings, except anti-Gray, are able to provide better performance then the Larsson labeling. The axis-aligned splitting method is able to provide the best performance as can be seen in Figure 9. Perhaps even more notable is that when this technique is a applied for a limited depth. Namely until depth of 4 instead of 8, it already provides significant performance improvements when compared to the labeling as proposed by Larsson. It should be noted that comparison against the Larsson labeling shows unrealistic large improvements. Since Larsson labeling is based on so called natural indexing of the constellation. Which is not optimized for BER performance. Furthermore no comparison is made against a labeling produced by the BSA algorithm on this constellation.

4.2 Performance over BICM

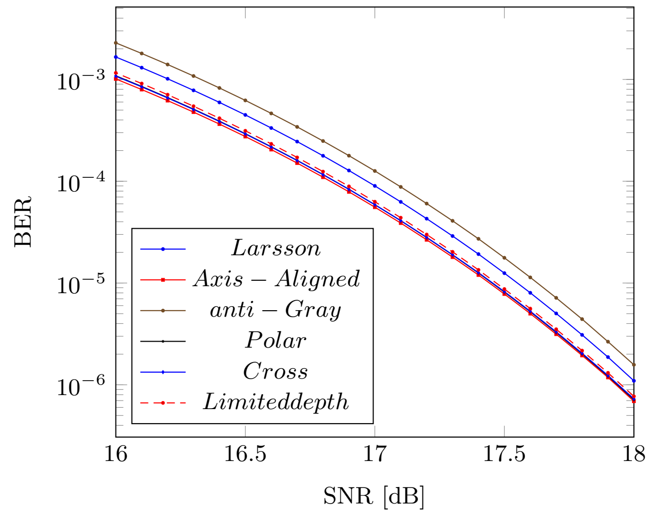

Figure 8. Comparison of various labeling methods

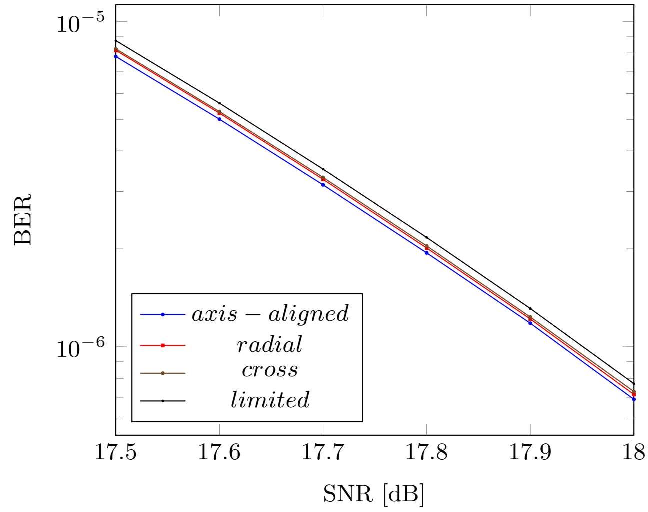

Figure 9. Comparison of splitting strategies, over AWGN channel

As can been seen in Figure 10, the Gray-code based labeling performed worse then the standardised labeling of DVB-S2. This can be explained due to lack of code diversity of the Gray-code based method. The anti-Gray code based method was supposed to have better performance in the high SNR region. However, as can be seen in Figure 10, it has a higher BER for a given SNR then even the Gray labeling. This suggests that the anti-Gray labeling method as proposed in Section 3.3 is not able to create a high code diversity. The results obtained by Xiang et al. in [37] were replicated. Suggesting that this is not an error from the simulator. The point at which the Gray coded labeled line drops down to a BER of 10^{-5} is at 14.5dB, one decibel later than the standard DVB-S2 labeling. However, for the low SNR region, the Gray-coding offers better performance over the BICM-ID channel. This can be seen in Figure 11. Now although the BER at this SNR is rather high, it does show that the ideal labeling in BICM-ID is dependant on the SNR received. Furthermore it shows that our algorithm is able to produce an effective quasi-Gray code mapping for the DVB-S2 mapping.

4.3 Asymptotic Analyses of the Proposed Algorithm

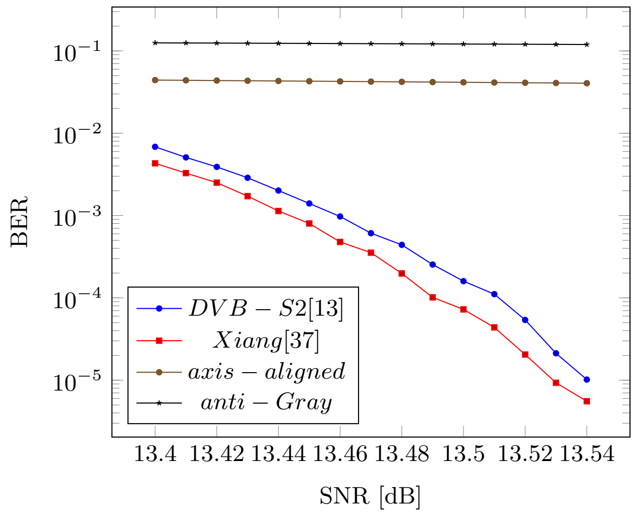

Figure 10. BER after LDPC decoding on DVB-S2 constellation, high SNR range

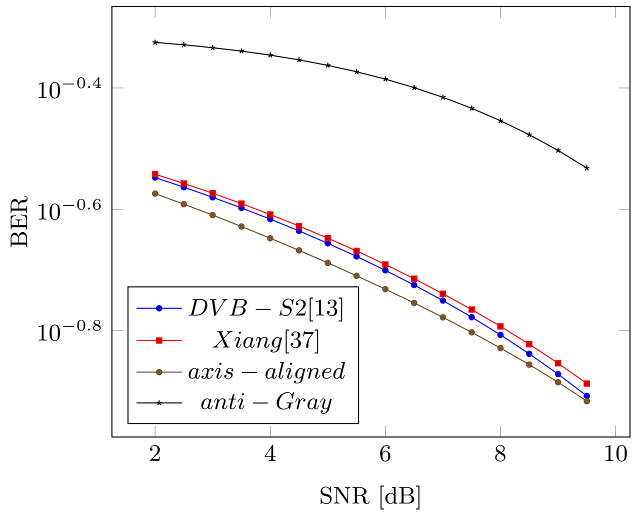

Figure 11. BER after LDPC decoding on DVB-S2 constellation, low SNR range

The proposed method definition consists of a recursive definition, making it especially suitable for asymptotic complexity analysis by means of the master theorem for divide-and-conquer recurrence [35]. Since the proposed method makes use of a recursive binary subdivision algorithm, we can split the problem in two sub problems of half the size each. Furthermore the cost of this splitting is solely based on the cost of deciding one which side of the splitting planes a constellation point belongs, currently implemented by means of list sorting, instead of a median finding algorithm due to greater flexibility of the former. The sorting algorithm used is C++ *sort()*, which for GCC compiler is the introsort algorithm [17]. Since the introsort based splitting step has a complexity of $\mathcal{O}(n \log n)$, therefore the complexity recurrence relation becomes:

$$T(N) = 2T\left(\frac{N}{2}\right) + \mathcal{O}(n \log n) \tag{3}$$

According to the master theorem this becomes a complexity class of

$$T(N) = \mathcal{O}\Big(n(\log n)^2\Big) \tag{4}$$

We can show that this complexity is less then quadratic in the problem size by simply noting that

$$\lim_{n\to\infty} \frac{(\log n)^2}{n} = \lim_{n\to\infty} \frac{\frac{2}{n}\log n}{1} = 0 \tag{5}$$

Hence the proposed method has a polynomial asymptotic complexity. Due to this low complexity, the labeling of a constellation of size 65356, only took 20 seconds. This means that this method is able to provide a labeling for even the largest constellation sizes. ADSL-2 for example can use constellations as large as 32768 points [22].

4.4 Performance improvement obtained by hardware acceleration

The transition probabilities were generated without the use of hardware acceleration as described in section 3.4.2, however since these only need to be generated once per constellation and SNR value, this was not a problem within the time constraints. The generation of a SNR values from -5 to 35dB in 0.1dB increments for constellation size of 256 points ended up taking 12 hours, on a simple server using only a single core.

The processing of the transition probabilities, which was GPU-accelerated using CUDA, ended up needing to spend 700ms for all 1200 possible combinations of 3 different mappings, and 400 SNR values. Of these 700ms, 400ms were spend on system time, such as file IO. Based on extrapolation of the run time, obtaining the Bit Error rate of these labelings using Monte-Carlo based methods would take 10 minutes to achieve the same results. Hence the GPU-accelerated approach performed two orders of magnitude faster.

4.5 Implementation in current hardware

Since the proposed method restricts itself to only relabeling of the constellation nodes, it can be implemented by means of a simple hardware lookup table of only 256 bytes for a constellation size of 256. This will fit in most telecommunication systems.

Since our labeling method is based on the KD-tree algorithm which allows efficient nearest neighbour search, our algorithm can be tightly integrated with the decoding hardware in simple receivers. For more complex receivers which use log-likelihood decoding and/or iterative decoding and demapping techniques. The implementation cost of these techniques will greatly exceed the cost of the relabeling which is required by our method.

5. DISCUSSION & FUTURE WORK

The proposed technique is trivial to extend to higher dimensional signal constellations. Higher dimensional refers to more orthogonal signals used for the modulation. Performance in that case will be more limited due to the curse of dimensionality. Namely that the amount of neighbouring nodes will increase, but that the amount of codes with low Hamming distance is limited.

5.1 Splitting strategies

One key observation to be made is that the best performing constellation splitting algorithm is the axis-aligned algorithm, and not the polar coordinates based one. The polar coordinates based approach should generate Gray coded rings. This suggests that constructing mappings were the rings are Gray-coded as done in [25] is not always the optimal choice. The less then expected performance of polar coordinates based splitting method is probably dependant on the spacing of the rings in APSK. For closely spaced rings, the Hamming distance with inner and outer rings becomes more important. Furthermore, the ring based splitting structure has a rather large contact circumference for the inner area containing the nodes. It is this circumference where the bit flipping occurs. Since on the other side of the division plane are the codes with one or more bits flipped. Whereas for the axis-aligned method, it produces rectangular areas containing similar bit patterns, having less circumference means less symbols which are close to the edge were bit flips occur. This grouping of bits in square regions is favorable over grouping in rings with a larger surface area, increasing the average distance between the labels with small Hamming distance.

5.2 Bit-interleaved coded modulation

Due to time constraints this research only considered the coded performance under LDPC, and only one particular instantiation of the LDPC code. Which is a limiting assumption since it is shown by Wesel et al. in [36] the best performing labeling is depended on the error correction used. Requiring different labeling strategies if the encoder trellis has parallel branches. Furthermore as already seen in Section 4.2 the best performing labeling is dependant on the signal-to-noise ratio that is being used.

5.3 Multiple-Input, Multiple-Output

A more interesting area of exploration is the non-coherent, Multiple-Input Multiple-Output systems (MIMO) channel. Were in the case of diversity coding using Space-Time Block codes (STBC) the optimal labeling is known for a 2 transmitter system to be the Alamouti code. Alamouti code only exist for a 2 transmitter system. For a system which has more senders and receiver assigning labeling becomes more complex.

The Alamouti code can be seen as a particular instance of the Grassmannian constellation. It is possible to construct the Grassmannian manifold for higher order STBC. Grassmannian constellation that mimic the isotropical distribution offer good performance when limited channel state information is available. Quasi-Gray labeling of the Grassmannian constellation as proposed by Colman et al. in [8] performed very well, both in uncoded situation and BICM with iterative decoding and demapping situations. The technique as proposed relies on mapping the Grasmannian constellation to a hypothetical constellation which is assumed to be perfectly Gray-labeled. Optimal matching is too computationally expensive, so a sub optimal mapping is used. A new technique could focus on extending the techniques as proposed in this paper to the Grassmannian manifold, a suggested area of work is by means of embedding the Grassmannian manifold for the constellation in Plücker coordinates. In this coordinate system the labelings can be created using the labeling technique presented in this paper. Subsequently the constellation points are mapped back to the Grassmannian manifold and used for modulation.

5.4 Constellation modulation

Constellation modulation is a technique proposed by Dash et al. in [9] which can improve spectral efficiency by as much as 33 procent. The technique works in the a very low signal-to-noise ratio regime, as low as 0.7dB. The axis-aligned Gray-code labeling technique proposed in this paper offered promising results at these low signal-to-noise ratios.

5.5 Optimality of anti-Gray code

As can be seen in Figure 10 and Figure 11, the anti-Gray code based labeling under BICM-ID error correction. performs worse than the Gray-code labeling for both low and high signal-to-noise ratios. From this we conclude that the anti-Gray code as created does not have a high code diversity, which is required for good performance under BICM-ID. This can be explained as being a side effect of the rather naïve approach of generating the anti-Gray code labelings. Being based directly on the gray-Code labels. This approach does not work because gray-Code and anti-Gray behave differently when considering a larger neighbourhood.

Consider the QAM constellation which is Gray coded. Each label has a Hamming distance of one with its orthogonal neighbors, and a Hamming distance of 2 with its neighbors. When the same experiment is done for an anti-Gray coded situation, The anti-Gray code generator has a characteristics where, the secondary neighbourhood is forced to have an undesirable low Hamming distance. The secondary neighbourhood cannot possibly have a high Hamming distance. This can be seen by a simple illustration: Suppose point A has code 0000, and point B is 1111, the highest possible Hamming distance. Then there is no way for a third point, point C. To have a high Hamming distance to both point A and point B. Hence anti-Gray code behaves different than Gray-code constellations. The authors believes that the creation of anti-Gray labeling could be improved by changing the assignment of labels to the tree to resemble the set portioning method.

5.6 Non equiprobability Constellation Points

The proposed methods assume equiprobability of each constellation point, which limits how close the constellation can approach a Gaussian noise distribution, which is needed to approach the Shannon capacity [15]. Larsson et al. showed how the GAM constellation can be used for non uniform disk shaping, and non equal probability of symbol transmission [26]. In order to improve shaping gain. Further research could extend this labeling method to non equal probability constellations.

5.7 Constellation sizes

Our method relies on binary splitting of the constellation points requiring the constellation to be of size 2^N, extensions could be make to allow this technique to be used on other constellation sizes, e.g. by changing one or more of the two-way splits to a three-way split.

6. CONCLUSION

Constellation labeling is a domain of telecommunication that allows performance improvements without increasing the complexity of the encoder or the decoder of the communication system.

In this research a rule-based method was realized which can efficiently calculate a quasi-Gray code constellation labeling. Quasi-Gray labelings are very important, because for a diverse range of applications, Gray-code based labelings offer the best performance. For example the best performance in phase shift keying, and rectangular constellations [2]. Furthermore Gray-code based labels offered the best performance in Non-Orthogonal Multiple Access Without SIC, as noted by Shieth et al. in [31]. This algorithm gives a labeling which improves the performance by 0.1dB to 0.2dB over AWGN channels compared to the labeling proposed by Larsson. This algorithm provides the starting point for future research in the creation of rule-based set-partitioning based constellation labeling for arbitrary constellations. Due to its low polynomial complexity of $\mathcal{O}(n \log^2(n))$, it can label large constellations. This enables the labeling of constellations which are too complex to approximate the optimum with the BSa algorithm. Furthermore it is shown that a reduced depth version already can provide significant performance gains.

The proposed method was also able to generate a labeling for DVB-S2 which performs better than the standard labeling for very low SNR values in BICM-ID channel.

A way to generate anti-Gray code labelings is explored, however the method used to generate them did not give the desired improvements.

7. ACKNOWLEDGMENTS

The author of this paper would like to thank Rolf van Kleef and Peter van Dijk, for repeated proofreading of this paper, and their moral support offered during the creation of this paper. Rolf van Kleef also donated spare CPU-cycles of his server for extended periods of time.

8. REFERENCES

[1] E. Agrell, J. Lassing, E. G. Strom, and T. Ottosson. On the optimality of the binary reflected gray code. *IEEE Transactions on Information Theory*, 50(12):3170–3182, Dec 2004.

[2] E. Agrell, J. Lassing, E. G. Strom, and T. Ottosson. Gray coding for multilevel constellations in gaussian noise. *IEEE Transactions on Information Theory*, 53(1):224–235, Jan 2007.

[3] A. Alvarado. On bit-interleaved coded modulation with qam constellations, Jun 2019.

[4] M. Anedda, A. Meloni, and M. Murroni. 64-apsk constellation and mapping optimization for satellite broadcasting using genetic algorithms. *IEEE Transactions on Broadcasting*, 62(1):1–9, Mar 2016.

[5] E. Biglieri, G. Caire, and G. Taricco. Coding for the fading channel: a survey. *Signal Processing*, 80(7):1135 – 1148, 2000.

[6] G. Caire, G. Taricco, and E. Biglieri. Bit-interleaved coded modulation. *IEEE Transactions on Information Theory*, 44(3):927–946, May 1998.

[7] X. Chen. *Error-Control Coding for Data Networks.* Kluwer Academic Publishers, Norwell, MA, USA, 1999.

[8] G. W. K. Colman, R. H. Gohary, M. A. El-Azizy, T. J. Willink, and T. N. Davidson. Quasi-gray labelling for grassmannian constellations. *IEEE Transactions on Wireless Communications*, 10(2):626–636, Feb 2011.

[9] S. S. Dash, F. Pythoud, D. Hillerkuss, B. Baeuerle, A. Josten, P. Leuchtmann, and J. Leuthold. Constellation modulation – an approach to increase spectral efficiency. *Opt. Express*, 25(14):16310–16331, Jul 2017.

[10] R. De Gaudenzi, A. Guillen i Fabregas, and A. Martinez. Performance analysis of turbo-coded apsk modulations over nonlinear satellite channels. *IEEE Transactions on Wireless Communications*, 5(9):2396–2407, Sep. 2006.

[11] L. Dolecek, Z. Zhang, M. Wainwright, V. Anantharam, and B. Nikolic. Evaluation of the low frame error rate performance of ldpc codes using

importance sampling. In *2007 IEEE Information Theory Workshop*, pages 202–207, Sep. 2007.

[12] Donald E. knuth. *The art of computer programming.* Addison Wesley, 2002.

[13] DVB Project. Digital Video Broadcasting (DVB);Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications;. https://www.etsi.org/deliver/etsi_en/302300_302399 /30230701/01.04.01_60/en_30230701v010401p.pdf, 2014.

[14] M. Ferrari and S. Bellini. Importance sampling simulation of concatenated block codes. *IEE Proceedings - Communications*, 147(5):245–251, Oct 2000.

[15] G. D. Forney. Trellis shaping. *IEEE Transactions on Information Theory*, 38(2):281–300, Mar 1992.

[16] G. D. Forney and L. . Wei. Multidimensional constellations. i. introduction, figures of merit, and generalized cross constellations. *IEEE Journal on Selected Areas in Communications*, 7(6):877–892, Aug 1989.

[17] GNU project. Sorting Algorithms. https://gcc.gnu.org/onlinedocs/libstdc++/libstdc++-html-USERS-4.4/a01027.html, Apr 2009.

[18] A. Grami. Chapter 7 - passband digital transmission. In A. Grami, editor, *Introduction to Digital Communications*, pages 299 – 355. Academic Press, Boston, 2016.

[19] S. H. Han, J. Cioffi, and J. Lee. On the use of hexagonal constellation for peak-to-average power ratio reduction of an odfm signal. *Wireless Communications, IEEE Transactions on*, 7:781 – 786, Apr 2008.

[20] W. Han, S. Le Goff, B. S. Sharif, and A. Al-Dweik. Power-efficient irregular repeat-accumulate encoded bicm-id for 16-ary signal constellations. *JCM*, 6:319–329, Jul 2011.

[21] R. Holzlohner, A. Mahadevan, C. R. Menyuk, J. M. Morris, and J. Zweck. Evaluation of the very low ber of fec codes using dual adaptive importance sampling. *IEEE Communications Letters*, 9(2):163–165, Feb 2005.

[22] International Yelecommunication Union. G.992.3 : Asymmetric digital subscriber line transceivers 2 (ADSL2). https://www.itu.int/rec/T-REC-G.992.3-200904-I, Apr 2009.

[23] Jet propulsion laboratory, NASA. Voyager spacecraft. https://voyager.jpl.nasa.gov/mission/spacecraft/, May 2019.

[24] Joseph I. Statman Charles D. Edwards, Jet propulsion laboratory, NASA. Coding, Modulation, and Relays for Deep Space Communication Mars Rovers Case Study. https://trs.jpl.nasa.gov/bitstream/handle/2014/39033/04-2151.pdf, Sep 2004.

[25] F. Kayhan and G. Montorsi. Constellation design for transmission over nonlinear satellite channels. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 3401–3406, Dec 2012.

[26] P. Larsson. Golden angle modulation. *IEEE Wireless Communications Letters*, 7(1):98–101, Feb 2018.

[27] MacGregor S. Reid. *Low-Noise Systems in the Deep Space Network*. Wiley publications, 2008.

[28] V. Nandal and S. Kumar. A comparative analysis of bit error rate of different symbol mapping schemes for bicm-id on different fading channels. *International Journal of Pure and Applied Mathematics*, 117:497–512, dec 2017.

[29] H. M. Navazi and M. J. Hossain. Novel method for multi-dimensional mapping of higher order modulations for BICM-ID over rayleigh fading channels. *CoRR*, abs/1807.04854, 2018.

[30] F. Schreckenbach, N. Gortz, J. Hagenauer, and G. Bauch. Optimized symbol mappings for bit-interleaved coded modulation with iterative decoding. In *GLOBECOM '03. IEEE Global Telecommunications Conference (IEEE Cat. No.03CH37489)*, volume 6, pages 3316–3320 vol.6, Dec 2003.

[31] S. Shieh, C. Lin, Y. Huang, and C. Wang. On gray labeling for downlink non-orthogonal multiple access without sic. *IEEE Communications Letters*, 20(9):1721–1724, Sep. 2016.

[32] J. Smith. Odd-bit quadrature amplitude-shift keying. *IEEE Transactions on Communications*, 23(3):385–389, Mar 1975.

[33] C. Stierstorfer and R. F. H. Fischer. (gray) mappings for bit-interleaved coded modulation. In *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, pages 1703–1707, April 2007.

[34] L. Szczecinski, S. Aissa, C. González, and M. Bacic. Exact evaluation of bit- and symbol-error rates for arbitrary two-dimensional modulation and nonuniform signaling in awgn channel. *Communications, IEEE Transactions on*, 54:956 – 956, Jun 2006.

[35] R. Verma. A general method and a master theorem for divide-and-conquer recurrences with applications. *Journal of Algorithms*, 16(1):67 – 79, 1994.

[36] R. D. Wesel, Xueting Liu, J. M. Cioffi, and C. Komninakis. Constellation labeling for linear encoders. *IEEE Transactions on Information Theory*, 47(6):2417–2431, Sep. 2001.

[37] X. Xiang, Z. Mo, Z. Wang, K. Pham, E. Blasch, and G. Chen. Constellation labeling optimization for bit-interleaved coded apsk. In *Proceedings Volume 9838, Sensors and Systems for Space Applications IX*, page 98380Q, May 2016.

[38] Xiaodong Li, A. Chindapol, and J. A. Ritcey. Bit-interleaved coded modulation with iterative decoding and 8 psk signaling. *IEEE Transactions on Communications*, 50(8):1250–1257, Aug 2002.

[39] T. Yoshida, M. Karlsson, and E. Agrell. Performance metrics for systems with soft-decision fec and probabilistic shaping. *IEEE Photonics Technology Letters*, 29(23):2111–2114, Dec 2017.

[40] Yuheng Huang and J. A. Ritcey. Optimal constellation labeling for iteratively decoded bit-interleaved space-time coded modulation. *IEEE Transactions on Information Theory*, 51(5):1865–1871, May 2005.

APPENDIX

A. CODE FOR LABELING OF THE TREE

```
treeNode* buildTree(treeNode nodes[], int depth, bool invertI, bool invertQ) {
    size_t size = CONST_SIZE >> depth;
    size_t halfWayIndex = (size - 1) / 2;

    treeNode *node = new treeNode();

    if (depth % 2 == 0) {
        std::sort(nodes, nodes + size, &compareI);
    } else {
        std::sort(nodes, nodes + size, &compareQ);
    }

    if (size == 1) {
        node->leave[0] = nullptr;
        node->leave[1] = nullptr;
        return &nodes[0];
    }

    treeNode* leftNodes  = nodes;
    treeNode* rightNodes = nodes + halfWayIndex + 1;



    if (depth % 2 == 0) {
        if (invertI) {
            node->leave[1] = buildTree(leftNodes,  depth + 1, invertI, invertQ);
            node->leave[0] = buildTree(rightNodes, depth + 1, !invertI, invertQ);
        } else {
            node->leave[0] = buildTree(leftNodes,  depth + 1, !invertI, invertQ);
            node->leave[1] = buildTree(rightNodes, depth + 1, invertI, invertQ);
        }
    } else {
        if (invertQ) {
            node->leave[1] = buildTree(leftNodes,  depth + 1, invertI, invertQ);
            node->leave[0] = buildTree(rightNodes, depth + 1, invertI, !invertQ);
        } else {
            node->leave[0] = buildTree(leftNodes,  depth + 1, invertI, !invertQ);
            node->leave[1] = buildTree(rightNodes, depth + 1, invertI, invertQ);
        }
    }
    return node;
}

void numberTree(treeNode* node) {
    if (node->leave[0]) {
        node->leave[0]->index = ((node->index << 1) | 0);
        numberTree(node->leave[0]);
    }

    if (node->leave[1]) {
        node->leave[1]->index = ((node->index << 1) | 1);
        numberTree(node->leave[1]);
    }
}
```