REPEATED RECORDING FOR AN OPTICAL DISC

F.M.J. Willems* and A.J. Vinck*

We describe the repeated recording model for optical discs and design three codes that can be used in order to store information in a reliable way. The third code has a rate of 0.517 bit per spot.

I. INTRODUCTION

On an optical disc we can record information if we use a laser and an electromagnet. The laser heats up a spot on the disc and on this spot either a 0 or a 1 is stored, depending on the orientation of the magnetic field generated by the electromagnet. We can record binary information on such a disc if we properly reverse the current through the coil of the magnet. An optical effect makes it possible to read the 0-'s and 1-'s on the disc, and in this way the recorded information can be reproduced.

If we want to store on the disc in a short time a huge amount of information, the inductivity of the coil of the electromagnet will prevent us from reversing the current. Therefore we propose the following strategy : A "new" disc contains only 0-'s. If we store information on the disc for the first time, we write only 1-'s and thus it is not necessary to reverse the current direction. Before storing information on the disc for the second time we reverse the current and now we write only 0-'s. The third time we write only 1-'s etc. For reasons of simplicity we assume that each time that we record, we rewrite the entire disc.

The interesting feature of the above strategy is that the states of the spots of (a part of) the disc are known to the writer, before

* F.M.J. Willems and A.J. Vinck are with the Department of Electrical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven.

he records new information on this (part of the) disc. This state information can be used by the writer. The reader has no information about previous states of the spots on the disc.

Our problem now is to find "good" codes for the above model. With good codes much information can be stored reliably on the disc. Remember that during odd cycles only 1-'s may be written and during even cycles only 0-'s.

II. SOME CODES

In this section we describe some simple codes. These codes all have the property that with probability 1 the reproduced information is equal to the stored information.

A. The blocklength of the first code is 2 spots. During the odd cycles 01, 10, or 11 is written, during the even cycles 00, 01, or 10. The information to be stored can take on 2 values, A and B. The tables below contain what is written, as a function of the previous states of the 2 spots and the information to be stored.

	00	01	10
A	01	01	10
B	11	11	11

odd cycles

	01	10	11
A	01	10	01
B	00	00	00

even cycles

If during an odd cycle an A has to be stored, 01 is written if the previous states were 00 or 01, and 10 is written if the previous states were 10. During an odd cycle we always write 11 if a B has to be stored. Reproducing the information contained in the 2 spots is simple, 01 and 10 corresponds to an A, 11 to a B. Note that during an odd cycle it is impossible to write a 0 if the previous state was 1, during an even cycle it is not possible to change a 0 into a 1. We remark that what we write during the odd cycles are the previous states for the even cycles etc.

Per 2 spots we can store 1 bit of information with the code described above. Therefore the rate of this code is 0.5 bit per spot.

B. The code described under A reads the previous states and writes an odd-weight codeword if an A has to be stored and an even-weight codeword if a B has to be stored. Note that the number of 10 combinations on the disc can never increase. Because we have assumed that a new disc contains only 0-'s, the combination 10 will never occur. Therefore we obtain the more simple tables below.

	00	01
A	01	01
B	11	11

odd cycles

	01	11
A	01	01
B	00	00

even cycles

It will be clear that this code does not need to read the previous states anymore. This makes the implementation a lot simpler. In fact this code writes during the odd cycles a 0 or 1 (depending on A or B) on spot 1 and always a 1 on spot 2, and during the even cycles always a 0 on spot 1 and a 0 or 1 (depending on A or B) on spot 2. From inspecting this code we see that it does not only write information but it also prepares spots for the next cycle. We therefore can call this code a time-share code. One can easily determine the spots that contain information and the spots that are prepared.

The rate of the code above is again 0.5 bit per spot, it is still the same code as described under A.

The question now arises whether or not such time-share codes are optimal. Under C we will describe a code that demonstrates that time-share codes are not optimal. In this code storage and preparation are present in a diffuse form.

C. The blocklength of the third code is 5 spots. During the odd cycles we write codewords with weight 3, 4, or 5, during the even cycles we write codewords with weight 0, 1, or 2. We describe only the code for

the odd cycles. Note that the codewords on the disc before writing, the previous states of the spots, have weight 0, 1, or 2.

01111	A set of codewords
10011	(a basic set).
11100	

Consider the above set of 3 codewords. All 3 codewords have weight not less than 3. We now want to know whether or not it is possible to write at least one of these 3 codewords when the previous codeword has weight not more than 2. More precisely, can we always choose one of these 3 codewords such that we do not have to change a 1 into a 0 (this is impossible during odd cycles)? The answer to this question is yes.

If the previous codeword contains a 1 on spot 1 the first word in the (basic) set (01111) can not be written since it has a 0 on spot 1. If the previous codeword contains a 1 on spot 2 or spot 3 we can not write the second word in the basic set. A previous codeword with a 1 on spot 4 or 5 eliminates the third codeword. Since a previous codeword has weight not more than 2, at most 2 codewords from the basic set are eliminated. The set contains 3 codewords and therefore at least 1 of these codewords can be written without changing a 1 into a 0.

We can say that the above set of three codewords "covers" all possible (weight not more than 2) previous codewords. If we now partition all words of weight not less than 3 in sets that cover all weight not more than 2 codewords, we obtain a code. As first set we take the basic set (01111,10011,11100). By permuting the columns of this basic set, we find 4 more sets of codewords that cover all previous codewords. Then only the word 11111 remains. This codeword however covers, on its own, all previous words and therefore forms a sixth set. With these 6 sets we can now store one out of six information symbols (A, B, C, D, E, and F) using 5 spots (see table on next page).

The rate of this code clearly is $\frac{1}{5}\log_2 6 = 0.517$ bit per spot.

A-set:	B-set:	C-set:
01111 (1)	10111 (2)	11011 (3)
10011 (23)	01101 (14)	01110 (15)
11100 (45)	11010 (35)	10101 (24)

D-set:	E-set:	F-set:
11101 (4)	11110 (5)	11111 (x)
01011 (13)	00111 (12)	
10110 (25)	11001 (34)	odd cycles

By inverting the code for the odd cycles we obtain the code for the even cycles. Note that the above code is rather good in the sense that the basic set and its 4 permutations exactly partition the set of words with weight 3 or 4. It is unknown whether or not there exist basic sets for higher blocklengths with the same property. Presently Erik Kwast is investigating this.

III. REMARKS

Using Shannon-theoretic arguments it is possible to show that rates higher than $\log_2((1+\sqrt{5})/2) = 0.694$ bits per spot can not be realized with reliable codes.

Furthermore it can be shown that rates arbitrarily close to 0.694 can be achieved with codes that have an arbitrily small but positive error probability. Recently John van Breemen has designed some codes of this type.

It should be noted that there is a close relationship between the configuration studied here and the Blackwell broadcast channel studied by multi-user information theorists.

ACKNOWLEDGEMENT

We thank Stan Baggen for proposing the problem investigated here to us. It is because of Krista that this contribution does not contain the proof of the fact that rates higher than 0.694 can not be realized.