Please refer this paper as http://www.r-ef.com/research/publications/codecs.pdf

Some geometric integer transformations

Pál Ruján

R-EF Research

Rujan Entwicklung und Forschung GmbH

Wintererstr. 47a, D-79104,

Freiburg, Germany *

(Dated: Feb. 28, 2015)

Abstract

Visual reference (REF) tags are q-state matrix codes designed for an optimal capture and decoding whilst the tagged objects or persons are moving relative to a standard digital camera. Let the maximal number of patterns which can be realized by such a tag be N. In this short note we show how to construct a set of distinct encoder-decoder pairs (codecs) by parametrizing the codecs with cyclic shifts. A set of codecs is distinct iff for each input number less than N the encoders generated visual patterns are all different. We prove that the maximal number of distinct codecs in the cyclic shift parametrization paradigm is N and that this is the largest possible distinct codec set.

* research@r-ef.com

I. INTRODUCTION

Visual reference tags are a new type of color matrix codes. Such tags are typically captured at a relatively large distance and several of them might be present on the same digital image. Hence, their size must remain relatively large while their storage capacity remains small. The information content carried by such tags changes typically between 10 to a few hundred bits. This raises the issue of how different application fields can use this technology without interfering with each other. Since the number of tags is relatively small, a "bandwidth assignment" type regulation does not scale well with increasing technological adoption and is too restrictive. Alternately, a content dependent or a geolocation based application segregation might or might not work, depending on the circumstances.

Consider for instance an application reading the geo-coordinates stored on a tag. Typically, such a tag will consist of a central hexagon surrounded by three subsequent layers and has a capacity of about 70 bits, when using six real colors + black and white. Another application might use the same type of tag for storing a web address and yet another one for storing street signs, intended as a "machine readable" shield for automatic car assistance systems. How can the application designers be sure that their capture software will not read a sign intended for another application and will be able to discard those tags, which are not intended for their application?

In fact, one can raise this issue for any other matrix code used in augmented (mixed) reality applications, where the storage capacity of the matrix code is restricted. Once a given technology becomes popular, many different applications might/will interfere with each other. The same problem exists in many other fields, including the problem of defining an universal, unique record id valid for all present and future database records. Note that a record id is usually a 64 bit integer, thus falling roughly within the category of our reference tags.

In this paper we show how to design simple distinct codecs. For these codecs no app-interference is possible, as long as the decoder is error free. We prove that the maximal number of possible distinct codecs corresponds to the tag's storage capacity. Our constructive solution generates the maximal number of distinct codecs and is hence optimal.

II. THE ENCODERS

Without loss of generality consider an integer number $n \in [0, N-1]$, where $C = \log_2 N$ is the storage capacity measured in bits. The application is designed such that this integer number contains the information content stored in a tag with storage capacity C. Sometimes we will also call N "storage capacity," instead of "the largest integer associated to the storage capacity". Hopefully this sloppy language will not create any confusion.

The integer n can be represented as a string of bytes or as a digital image (two-dimensional array), or any other entity storable in the memory of a computing device. Hence, without loss of generality, n is called the "the input message". We transform this message into another form better suited for a graphical display. This is usually a visual pattern we will call a reference tag, because it contains only a reference to the place where more information about the tagged object or person can be found. Once an image processor has located and evaluated a graphical tag from a digital image, this will be presented in a form accepted as input by a decoder, which restores then the original message. In this paper, we are considering only the transformation between the original message into a graphical pattern and back, the image processing aspects will be discussed in a separate publication (but see [1]).

In general terms, let us assume that the design is such that any integer input $n \in [0, N-1]$ will be mapped into one distinct color pattern, different from all other N patterns. At this stage we do not have to concretize how this happens. It suffices to point out that the transformation from an integer to a tag coloring is akin to representing the same number in two different number systems. Remember, a q-nary number representation is defined as

$$n = \sum_{i=0}^{M} a_i q^i, \text{ where } n \in [0, N-1] \tag{1}$$

As argued elsewhere [1], in order to encode an integer into a graphical pattern it is sometimes necessary to use a mixed number representation, in which every $i-th$ digit is coded as one of q_i-states ($N \leq \prod_{k=0}^{M} q_k$) :

$$n = \sum_{k=0}^{M} a_k Q_k, \text{ where } \; a_k \in [0, q_k - 1], \quad Q_k = \prod_{i=0}^{k-1} q_i \text{ for } k > 0 \text{ and } Q_0 = 1 \tag{2}$$

Consider a number between 0 and 99. In usual situations (decimal representation) this

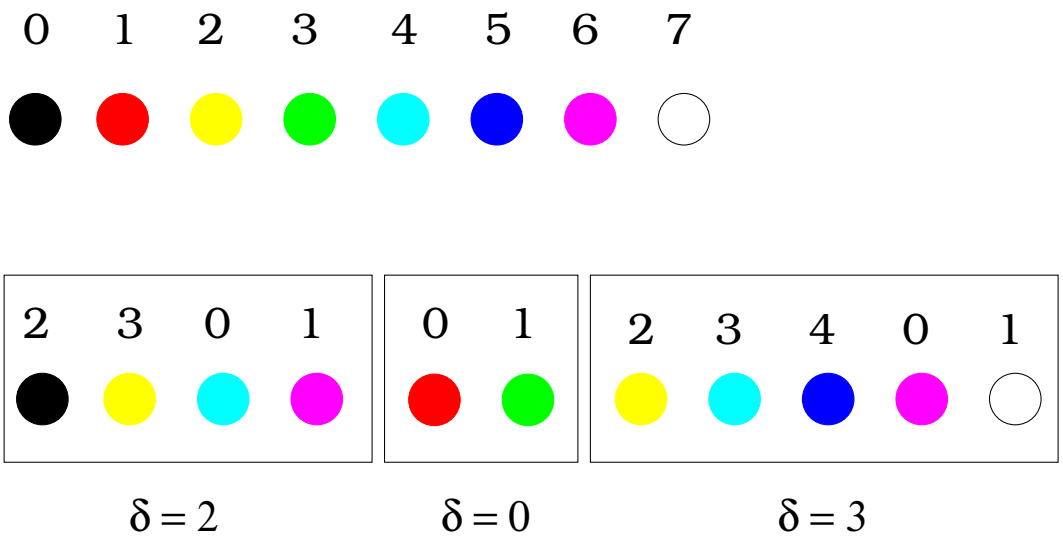

FIG. 1. *Upper part: ordering colors in lexicographic sRGB order. Bottom: coding colors to digits for given δ cyclic shift values. Some colors are absent due to geometric exclusion rules, the available ones are sRGB ordered, and counting the digits is cyclic, continues from left. The upper row shows the map to a_i coefficients after the δ right cyclic shifts are performed.*

is expressed with the help of two digits. From a geometric point of view each integer value corresponds then to the co-ordinates of a 10×10 grid, written as $x + y * 10$. However, the same value can be written as the co-ordinates of a $2 \times 5 \times 2 \times 5$ grid or a $4 \times 5 \times 5$ grid, each co-ordinate multiplying the volume of the corresponding subgrid.

As shown in Fig 1 one can associate a direct map between different colors or other visual symbols such that each digit corresponds to another member of the color set. Hence, once we have defined the sequence of the digits in the geometric representation of the reference tag, (which we call the "encoding path") we are free to define this map for each digit. This might sound at first strange but think about a cryptographic block coding method where for each letter in a word one redefines the alphabet, or for each digit of a large number one redefines the symbols $\{0, 1, \ldots 9\}$ arbitrarily. In fact, such a recoding would correspond to a cypher-block in the practice of cryptography ([2]). Next, let us ask the question:

How many such encoders exist?

To fix the parameters, consider integers of form $n \in [0, N-1]$ where the digits contain $\{q_0, q_1, \ldots q_M\}$ states, such that $n \leq N \leq Q_{M+1} = \prod_{k=0}^{M} q_k$. The number of permuting a set with q_k elements is $q_k!$, so that the number of choices when the encoding path is fixed is

$$Z(digits) = \prod_{k=0}^{M} q_k! \tag{3}$$

To better grasp this result, consider a decimal number n with 10 digits. Hence, $q_i = 10\ \forall i$ and $Z(10) = (10!)^{10} \simeq 10^{100}$ when using the first term of the Stirling formula. By comparison, the maximal value of n is only $N = 10^{10} - 1$. The possible number of encoding paths defining the ordering of the digits is $\frac{M!}{n(q_0)!n(q_1)!\ldots n(q_M)!}$, where $n(q)$ is the number of digits with the same number of states $q_k = q$ along the encoding path. Hence, one has:

$$Z = \frac{M!}{n(q_0)!n(q_1)!\ldots n(q_M)!} \prod_{k=0}^{M} q_k! \tag{4}$$

This is a huge number. However, we are not interested in all possible encoding schemes but only on **distinct** ones. We call a **distinct** codec set one with the following properties:

1. Every encoder-decoder pair recovers exactly all input messages in the full range $n \in [0, N-1]$,

2. Every single encoder-decoder pair (or codec) generates a different visual patterns for each input $n = 0, 1, \ldots N-1$. In other words, each codec maps uniquely n to a color pattern and back and all available patterns are used.

3. Encoders of a distinct codec set generate different reference visual patterns $\forall\ n \in [0, N-1]$, so that the patterns differ pairwise in at least one of their digits (or pattern cells).

III. THE CYCLIC SHIFT CODECS

We follow the ideas presented in [1] and define a very simple way of generating a set of codecs. Let us first assume that the used colors form an **ordered** set, they are ordered for instance according to the lexicographic order of their sRGB coordinates. In addition, let us assume that they are arranged on a circle, so that given q colors as $\{0, 1, \ldots q-1\}$, the color q is the same as color 0, etc. Such transformations are known in the literature as cyclic shifts.

Therefore, in order to define different encoders, we define for each encoding step involving a digit a shift variable, $\delta_k = \Delta_k \mod q_k$ and start counting the colors starting with the

$\delta_k - th$ one and then count them cyclically $\mod q_k$. The totality of shifts for all digits will define (in the same order as they are applied) an integer number of the same kind as the initial message and it is not difficult to see that its maximal value is also N, the number of available color patterns. The decoders will follow the same logic and will also depend on their predefined set of cyclic shifts.

Therefore, there are in total N possible codecs using cyclic shifts along a given encoding path.

We will now show that although the number of existing codecs is Z, the number of **distinct** codecs is only $N - 1$ and thus *the cyclic shift method provides the most general (optimal) realization of all distinct codecs.*

From the definition of distinct codecs it follows that there can be only N distinct encoders and decoders, in short *codecs*. The same applies for the cyclic shift codes, so they are a full representation of all possible distinct codec sets. Note that an encoder can output its set of cyclic shifts by simply encoding the input message $n = 0$ (zero).

A more interesting corollary is that even if we change the encoding path, the maximal number of distict encoders remains N. Therefore, changing the encoding path will not increase the number of usable, distinct codecs. However, it makes more difficult to 'crack' a hardwired encoder, because without knowing the encoding path outputing number zero does not provide much information about the initial message. A good choice of the encoding path might be useful for security purposes but cannot increase the distinct codec set's capacity.

IV. THE DECODERS: SOLVING THE MULTI-APP PROBLEM

We are now in the position to define a validation method recognizing the app for which a visual pattern is intended to. Consider two applications, each using a different set of cyclic shift codecs. In a practical situation these are two phone apps, capturing both the same visual reference tag. Let us assume that the tag has been encoded with the shift-parameters hardwired in the encoder of App_1, which are different from the encoder shifts used in App_2. The image is processed and both decoders will generate an integer number, n_1, respectively n_2. At this stage, the decoders have no idea whether this result is correct or not, except perhaps for some content dependent plausability checks. Therefore, in order to check the validity of the codec parameters (the foreign encoder had the same internal key as the own

decoder) one needs some extra information, whose maximal size equals that of the stored message. In some cases, a smaller key is also sufficient: in such cases one can generate a hash key or use a publicly known device number to identify the app task domain. A practical way is to use a strong error-correcting code, like a low parity Gallagher code with R=1/2. For a small number of detected errors it works like a powerful error-correcting code and in case the encoder and decoder are incompatible, it generates (because of the distinct propert) a massive amount of errors. These codecs are implemented before (and after) the integer-color code transformation and color-integer transformation in the decoder, respectively.

In very noisy conditions it is useful to introduce additional color-field redundance to help further the image processor identify correctly the color pattern. This type of error-correction must be well tuned to the already existing nearest-neighbor exclusion rules, which are redundant in their own. Assuming there are 2^{70} legal colorings, a capacity of 70 bits could be split into a 35 bits shift half and a 35 bit encoded message. Of course, one could partition the 70 bits also into a 32+38 bits, etc. Nota bene: the cyclic shifts used for encoding/decoding are easily implemented and already supported by hardware in many processors. A short and formal description of our results can be found in [3].

---

[1] P. Rujan: *Producing, Capturing, and Using Visual Identification Tags for Moving Objects*, provisional patent application dated 10.08.2010 US 13/206977, EP 11757789.0
`http://www.r-ef.com/research/publications/reftags_patent.pdf`

[2] B. Schneier: *Applied Cryptography*, John Wiley and Sons, Second Edition, 1996

[3] P. Rujan: *Block Cyphers, Keys, and abelian integer transformations*
`http://www.r-ef.com/research/publications/codecs_short.pdf`