Incremental Information Objects and Progressive Barcodes

Steven Simske, Hewlett-Packard Labs, 3404 E. Harmony Rd., MS 36, Fort Collins CO 80528, USA
Marie Vans, Hewlett-Packard Labs, 3404 E. Harmony Rd., MS 36, Fort Collins CO 80528, USA
Brad Loucks, Hewlett-Packard Labs, 3404 E. Harmony Rd., MS 36, Fort Collins CO 80528, USA

Abstract

In the fast-changing world of actionable printing, there has recently been a huge increase in the adoption of 2D barcodes for enterprise and consumer applications. The Data Matrix 2D barcode has become a primary carrier of supply chain information, most notably for track and trace. The QR (Quick Response) 2D barcode, meanwhile, has spread from Japan to the rest of the world, and is a standard means of connecting a barcode to a URL. We have previously developed a 3D (color + 2D) barcode that increases the barcode data capacity and provides branding possibilities. In this paper, we introduce the 4D barcode (color + 2D + time) as a means of using the same barcode location for multiple barcodes through time. This supports many enterprise workflows, including document lifecycles. In this paper, we consider the theory behind incremental information objects (IIOs), of which progressive barcodes are a key incarnation. We show that the set of progressive barcodes comprising a lifecycle can contain up to twice the data of the associated single-time 3D color barcode.

Introduction

In order to support packaging and document lifecycles such as track and trace and signatory workflows, respectively, we needed a printed mark that does not change size (take up more "real estate") as it has more workflow information written to it; that is, a printed mark that can be incrementally read and written to during various stages in its lifecycle. Because of the nascent ubiquity of high-resolution mobile cameras, the Data Matrix [1] and QR 2D barcodes have been adopted for many applications [2] that tie a physical object or location to electronic workflows (as simple as connecting to a URL, or as complicated as providing a log-in screen for member services). The groundwork has thus been laid for higher-density color barcodes – named 3D barcodes for their 2D layout and their third, color, dimension – such as the color tile [3] shown in Figure 1.

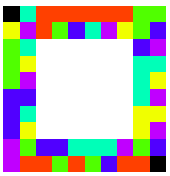

Figure 1*. Example of 3D (color) barcode, the color tile.*

Building on the color tile, we wanted to create a color tile that changes through time, so that the same barcode object can be used as an instance in a sequence of related objects and so empower complex, time-dependent workflows.

The Incremental Information Object (IIO)

An incremental information object (IIO) is an object that changes as a one-way function of its current state. Suppose we start with a simple binary sequence {000000000000} and then move to a next state through the replacement of four 0's by four 1's. Then, two allowable next states are {001010001100} and {100110000100}. In general, if there are N 0's left to be changed into 1's and M 1's added to the next state, then we can write N! / [M! (N-M)!] different next states, where ! is the factorial operator. For the IIO, once a 0 has been changed into a 1, it cannot change back into a 0. Thus, each successive state can be immediately compared to a previous state to see if it is logically a part of the same workflow. Figure 2 illustrates the progression for a three-state document workflow. Note that once a module has been turned dark it cannot be turned white later.

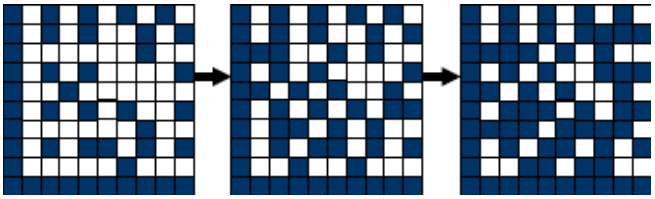

Figure 2*. Example of IIO as progressive barcode The barcode on the left indicates the first state of the workflow (e.g. it is affixed to the document when printed); the barcode in the middle indicates the second state (e.g. the document after signed and scanned, then re-printed); and the barcode on the right indicates the third and final state of the document (after signed and approved).*

The Progressive Barcode IIO

To generate a barcode representation of the IIO, in the above we can simply let 0=White and 1=Black, so that a black and white barcode like object can be an IIO. For higher density, we wanted a color IIO. For a color progressive barcode, the color transformation processes must be selected. Because of the reality of most printing – centered on the use of subtractive inks, cyan (C), magenta (M) and yellow (Y) – the color progression shown in Figure 3 is selected.

For that progression, a white tile can only progress to cyan, magenta or yellow; a cyan tile may only progress to blue or green; a magenta tile may only progress to blue or red; a yellow tile may only progress to green or red; and a red, green or blue tile may only progress to black. Since a tile may also stay in its current state for one of the allowable input values, it is possible that some tiles will stay white even while others progress fully to black.

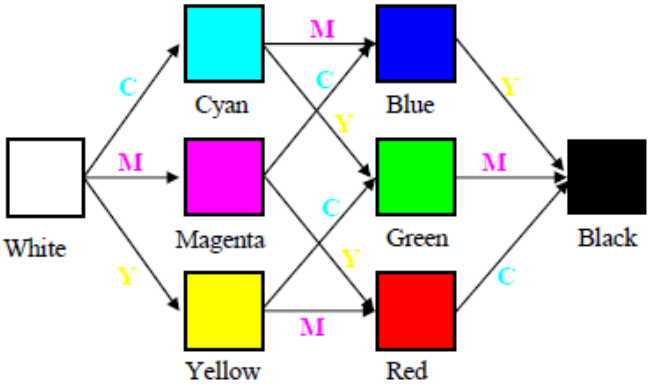

***Figure 3**. The basic lifecycle of a color tile, where the colors White, Cyan, Magenta, Yellow, Blue, Green, Red and Black are shorthanded as W, C, M, Y, B, G, R and K, respectively. Note that the tile may be written to three times in its lifecycle, and contain one of 8 states (3 bits). At the first stage, the W tiles can be overprinted with C, M or Y to create a C, M, or Y colored tiled, respectively (pretty obvious in hindsight—such is the nature of subtractive printing). At the second stage, the C tile can be overprinted with M or Y to create a B or G tile, respectively; the M tile can be overprinted with C or Y to create a B or R tile, respectively; and the Y tile can be overprinted with C or M to create a G or R tile, respectively. The (CMY) characters associated with the arrows indicate the additionally printed color required for the progression.*

An allowable set of color progressions from white to black using only the subtractive (CMY) and additive (red, green and blue, or RGB) primaries is possible, and this uses the same set of colors and color science (e.g. spectral pre-compensation) as the original 3D color tiles [3], as shown in Figure 3. Note that we will use the symbols {WCMYRGBK} throughout this document to represent, in order, White, Cyan, Magenta, Yellow, Red, Green, Blue and Black colors.

The overall "state" of an IIO is determined by the states of all of elements simultaneously. If, for example, we were to use a 64 payload element color tile such as shown in Figure 1, there are 2^{192} or more than 5.8 x 10^{57} possible states in just this small IIO (where there are 8 colors—WRGBCMYK—or 3 bits/payload element).

Data Carrying Capacity of the Progressive Barcode

For the purposes of determining the absolute data content of a color tile, we consider each color tile to be independent. We define the tile to be **n**-ary, where **n**=the number of colors allowed at each tile. Thus, there are $\log_2(n)/\log_2(2) = \log_2(n)$ bits at any stage. If **n**=2, as for 2D DataMatrix, QR, Aztec and similar binary 2D barcodes, then there is 1 bit per tile. For a color tile with six colors {RGBCMY}, there are 2.585 bits/tile. For seven colors, {RGBCMYK} or {RGBCMYW}, there are 2.807 bits/tile. Finally, if there are eight colors allowed {RGBCMYWK}, there are obviously 3.0 bits/tile exactly. This means that a color tile barcode that is X data tiles wide and Y data tiles high contains exactly $XY\log_2(n)$ bits.

However, this absolute density refers to a one-time use – or 3D barcode – implementation. In progressive applications, we wish to write numeric strings – preferably binary – to the progressive barcode to provide a (statistically) secure state-to-state transition.

Figure 3 illustrates the allowable transitions in the standard 8-color progressive barcode, catalogued in Table 1.

Current Color	Allowable Next States	Bits Supported
W	W, C, M, Y	2.0
C	C, B, G	1.585
M	M, R, B	1.585
Y	Y, G, R	1.585
R	R, K	1.0
G	G, K	1.0
B	B, K	1.0

***Table 1**. The current color (K not shown as it cannot progress and thus supports the writing of 0 bits), its allowable next states and the bits supported.*

Table 1 illustrates how binary, trinary and quarternary data can be written to color tiles with a current state in the set {WCMYRGB}. Since the (fully random) odds of a color tile remaining at the current state are 1/n for an n-ary system, then the number of bits at an n-ary stage are equal to $\log_2(n)\sum_{i=1}^{\infty}\left(\frac{1}{n}\right)^i$ $= \log_2(n)\sum_{i=1}^{\infty}\frac{1}{n^i}$.

Since $\sum_{i=1}^{\infty}\frac{1}{n^i}$ = 1/(n-1) for all n>1, we have a lifetime expected number of bits at each state of q-ary possibilities of $\log_2(q)\left(1+\frac{1}{q-1}\right) = \log_2(q)\left(\frac{q}{q-1}\right)$

Current Color	Allowable Next States	Lifetime Bits
W	W (0), C (1), M (2) , Y (3)	2.667
C	C (0), B (1), G (2)	2.377
M	M (0), R (1), B (2)	2.377
Y	Y (0), G (1), R (2)	2.377
R	R (0), K (1)	2.0
G	G (0), K (1)	2.0
B	B (0), K (1)	2.0

***Table 2**. The writing of allowable states to the progressive color barcode.*

As Table 2 shows, the expected lifetime of bits written rises above the bits available at one state – that is, $\log_2(q)$ – in direct proportion to the ratio $\left(\frac{q}{q-1}\right)$, where q > 1. That means that the greatest "gain" is obtained when q=2, and the expected number of bits supported by the tile is twice its *static carrying capacity* (2.0 bits expected lifecycle bit carrying capacity, and 1.0 bits at a given state). This is because when the entered bit stream carries a "0" in n-ary representation, the tile stays at the same location in its progression and so can be used to carry bits at the next stage.

Therefore, in any progression from W→{CMY}→{RGB}→K, the statistical, or expected, bit carrying capacity of the tile is 2.667 + 2.377 + 2.0 = 7.044 bits. This is a huge increase (135%) over the 3.0 bits carried by a static {WCMYRGBK} representation.

However, there is a problem. Mapping binary strings directly to trinary decisions, such as C→{CBG}, M→{MRB} and Y→{YRG}, as shown in Table 2, is messy and inexact. For example, suppose we have a 2x2 tiles structure currently consisting of the sequence {RCYW}. Then, we have 2x3x3x4=72 possible next states. We can map out the first 64 using 6 bits, but then the remaining 72-64=8 next states cannot be mapped to.

So, instead, we generally reduce trinary decisions to binary decisions upfront, and so gain a direct mapping between input binary strings and the progression of the IIO. Table 3 illustrates our implementation of this approach.

Current Color	Binary Allowable Next States	Lifetime Bits
W	W (00), C (01), M (10), Y (11)	2.667
C	C (0), B (1)	2.0
M	M (0), R(1)	2.0
Y	Y (0), G(1)	2.0
R	R (0), K (1)	2.0
G	G (0), K (1)	2.0
B	B (0), K (1)	2.0

Table3*. The current color and the binary representation of the allowable next states.*

In Table 3, the mapping of W, R, G and B remains the same, as these exactly support two bits {W} or 1 bit {RGB}. For W→{WCMY} we let the 2-bit binary string S map as follows:

if S = 00, then W→W
if S = 01, then W→C
if S = 10, then W→M
if S = 11, then W→Y

Similarly, we map the following for R, G and B with regard to a 1-bit binary string S:

if S = 0, then R→R
if S = 1, then R→K

if S = 0, then G→G
if S = 1, then G→K

if S = 0, then B→B
if S = 1, then B→K

The trinary decision, however, is retrofitted into a binary decision, such that:

if S = 0, then C→C
if S = 1, then C→B

if S = 0, then M→M
if S = 1, then M→R

if S = 0, then Y→Y
if S = 1, then Y→G

That is, transitions from C→G, M→B and Y→R are not allowed. Note that {RGB} are still, over time, equally represented since (C+M+Y) still maps to (R+G+B). Here, the lifetime statistical, or expected, bit carrying capacity of the tile is 2.667 + 2.0 + 2.0 = 6.667 bits. This is still a huge increase (122%) over the 3.0 bits carried by a static {WCMYRGBK} representation, and only a 5.4% reduction from the trinary case outlined in Tables 1-2.

We feel the direct mapping between binary strings makes this modest (5.4%) reduction well worth it.

References

[1] International Standard ISO/IEC 16022:2006(E), Second edition 2006-09-15, "Information technology – Automatic identification and data capture techniques – Data Matrix bar code symbology specification," 142 pp., 2006.

[2] http://www.redlaser.com, last accessed 2 February 2012.

[3] Simske SJ, Sturgill M, Aronoff JS: Effect of copying and restoration on color barcode payload density. Proc. ACM DocEng 2009 127-130, 2009.

Author Biography

Steve Simske is an HP Fellow and the Director and Chief Technologist of the Document Ecosystem portfolio in Hewlett-Packard Labs. Steve is currently on the IS&T Board. He is also an IS&T Fellow and a member of the World Economic Forum's Global Agenda Council on Illicit Trade. Steve has advanced degrees in Biomedical, Electrical and Aerospace Engineering, and has more than 50 granted US patents.