https://doi.org/10.2352/ISSN.2169-4451.2018.34.192

Extending the Reach of a Barcode-Based Imaging System

Matthew Gaubatz[1], Marie Vans[2], Steven Simske[3]
[1]HP, Inc., Seattle, WA, [2]HP, Inc., Fort Collins, CO, [3]Colorado State University, Fort Collins, CO, USA

Abstract

An imaging ecosystem, such as a supply chain or manufacturing line requires imaging devices to interact with 2- and 3-D objects. In such a setting, different services are exposed to and/or provided by the objects via image capture and analysis: examples include track-and-trace, shopping assistance, recall administration, access to customer service and rights management. For ease-of-use and ease-of-adoption, the best solutions allow a wide range of objects to be interrogated by a range of devices. Progressive barcodes offer a convenient scheme to image a dynamic object throughout an ecosystem than can extend into the actual production of an object or part. This paper investigates a framework for extending the reach of an image ecosystem with three components: Object-centric functionalities invoked via barcode scanning operations, hybridization of mobile imaging and desktop browsing environments, and a compact, evolving representation of data that can be rendered and interpreted on 3D surfaces.

1. Introduction

Traditional track-and-trace solutions enable the ability to interrogate a specific item or object as it flows through a supply chain. This functionality serves various entities in a supply chain, including manufactures, distributors, wholesalers, consumers, shippers, customs agents and security personnel. As additive manufacturing applications become more prominent, the opportunity arises to extend the ability to track an item through its own production process. Barcoding techniques [1,2] are a prominent component of many tracking solutions. *Progressive* barcodes [3,4] are a specific class of essentially 4-D compact barcoding solutions that are particularly well suited for tracking multi-stage processes for smaller objects.

Recent advances in web-based imaging APIs [5-7] have expanded opportunities to support imaging applications across a wide range of devices. At the same time, improvements in tooling to compile C++ code into java script/WebAssembly applications has enabled the ability to move heavier and more complicated processing into client applications [8,9]; an increasing number of C/C++ libraries are becoming accessible in a web-only context. Furthermore, the ability to draw on resources, such as GPU computing, is being increasingly exposed through calls invoked from the web [10,11]. Due to these changes, the presence of barcodes and other like markings increases the ease with which object-centric applications can developed and deployed.

This paper investigates a web-enabled object-centric imaging scheme for providing track-and-trace and other services throughout an ecosystem in which an object is produced and used. The approach is easily accessible from any web-connected device, and while progressive barcodes provide convenient scanning options for mobile devices [12], it also benefits from the ease with which a (desktop) website may be developed, without, strictly speaking, requiring any deployment overhead, i.e., it can be used in a completely local setting. Furthermore, the scanning application can leverage GPU computing techniques exposed via the desktop browser, which eases the development burden and extends the computational range of the imaging device. In this setting progressive barcodes are beneficial because they are dense, dynamic, and are convenient for tracking an object as it is being produced, as well as after it has been produced.

This paper is organized as follows. The progressive tracking technology is discussed in relation to additive manufacturing applications in section 2. A major challenge in creating a useful solution is the ability to easily deploy components that can image true 3D objects; the prototype system used for that purpose is described in section 3. Scanning performance is discussed in section 4, and section 5 provides resulting implications and offers concluding remarks.

2. Tracking Objects and Components

Progressive barcodes provide the ability to track an object throughout a multi-stage process. Key properties of these codes include (1) implementation as a layer of information on top of existing barcode standards, (2) the ability to estimate probabilities at each step in the progression of the process, and (3) conveniently readable across a range of devices. Figure 1 illustrates a progression of states, represented by colors; in this example states are represented by colors from any one of the following sets: {white}, {cyan, magenta, yellow}, {red, green, blue}, {black}. These hues represent colors obtain via overprinting. In a multi-layer rendering process, there is greater freedom in selecting colors and/or properties to represent states, as well as connection between states in adjacent stages.

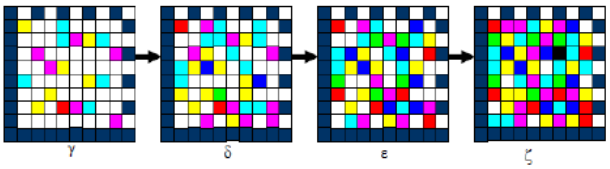

Figure 1. *Example progression of states through four stages. In an overprinting system, colors may advance from white to a primary ink color, then to a composite color and finally to black, in that order. Systems capable of printing new colors in each layer offer increased flexibility.*

Probabilities describing the stages, given by P_i, can be bounded as follows.

$$\#(\text{initial unwritten bits})!/\#(\text{final unwritten bits})! \geq \Pi i \; Pi. \qquad (1)$$

Using this formulation, parts that have been diverted and/or obtained through illicit activity can be traced, probabilistically, to a particular stage in the lifecycle of the item. One of the benefits of using this approach for an additive manufacturing application is that when using rendering systems that can create color designs at each layer, the space of states and the associated probabilities can be generated with greater specificity.

A complete diagram representing a progression throughout and after the production of a part is given in Figure 2. The nesting property of the progressive codes can be applied to generate relationships between individual components that get assembled into a single part in the same way that individual parts are grouped together in a carton. At the same time, it can be used to reflect components that are assembled into a single object.

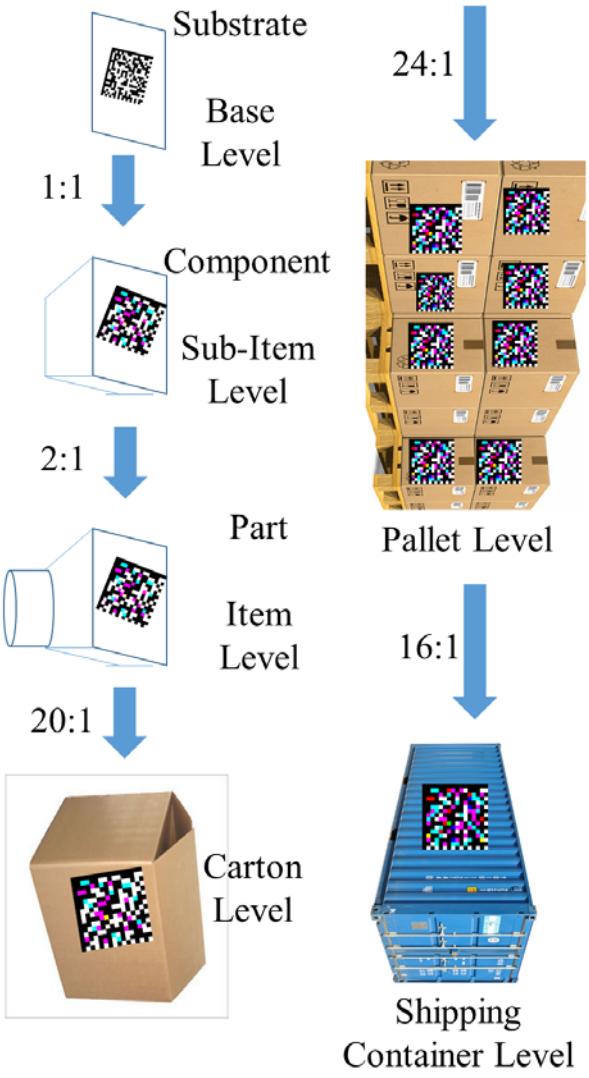

Figure 2. *Diagram illustrating example color QR code progression though production and distribution of an example part.*

3. Imaging System Overview

The proposed approach is designed to be convenient to implement and deploy. A prototype implementation was created using off-the-shelf components to generate barcode tracking measurements for codes rendered on top of non-planar objects. A simple web-page, accessible either from a desktop device or a mobile phone, was created to implement a baseline Data Matrix reader to detect the presence of the marking. The same toolchain was used to extend this approach to be able to interpret color barcode data, i.e., information that forms the progressive layer for the scheme illustrated in Fig. 1 and 2. A standard Android phone (Samsung Galaxy S6) was used as an imaging mechanism. The web application was deployed to be externally accessible, but also was tested locally on a PC in conjunction with some tools that allow mobile phone cameras to be accessed as PC peripherals [13,14], as well as with USB cameras.

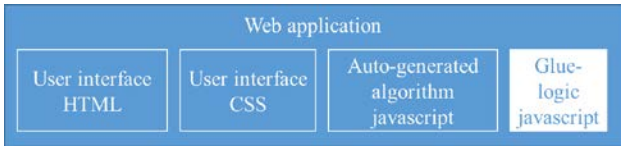

Figure 3. *Components of proposed scanning framework. The major manual development task focuses on assembling the glue-logic to drive the application; interface and algorithm code can be handled using known design principles and automated tooling.*

The design of the system is illustrated in Figure 3. The majority of the *total* development tasks required to implement this solution consist of using existing toolchains to transform popular pre-existing C++ libraries into java script, or to use well understood UI principles to create a usable scanning experience with minimal effort. Table 1 illustrates advantages and constraints associated with multiple scanning architectures that can be achieved with the exact same software. In one instance, the system can be run in within a mobile browser. In another, the same software provides a mechanism for a desktop or otherwise more powerful system to run the same operations, either with a dedicated USB camera device, or the presence of a mobile phone acting as such. Because the approach can be used from different mobile/computing devices, the same scheme can be deployed to a range of points throughout a given ecosystem.

Table 1. Imaging systems v. advantages and constraints.

Imaging setup	Advantages	Constraints
Mobile device only	- lightest hardware requirements - portable - video or still-capture	- internet required - UI limited to screen - indirect camera control
PC + mobile device	- no USB required - navigable UI - autofocus control - any OS	- FPS constrained by PC connection
PC + peripheral camera	- most control over imaging experience - navigable UI - any OS	- heaviest hardware requirement

One example imaging setup (PC + mobile device) is depicted in Figure 4. This example illustrates some of the advantages and disadvantages of the scanning system. In this setup, it is possible to move the mobile device anywhere in the scanning environment even to locations distant from the PC. The penalty for this flexibility is that the video feed displayed on the PC is transmitted over a wireless connection, which affects the framerate. For some types of printing systems, this constraint may preclude use for real-time monitoring, but for many 3D rendering systems, the same issue would not apply. The view of the substrate (in this case, a hybrid QR/Data Matrix code on a cylindrical surface) is easy to see on the computer, which can also adjust some of the imaging controls associated with the phone.

Figure 4. *Example of imaging setup using the proposed approach. The notebook has loaded a web-based application to drive an imaging device, which in this case is the phone in the scene. The video feed from the phone, viewable on the phone screen, is processed on the notebook as if the video was obtained from a computer peripheral.*

4. Scanning Evaluation and Discussion

A critical capability of an imaging system designed for tracking 2D and 3D objects is to scan codes on non-planar surfaces. There have been several recent studies devoted to analyzing and/or improving signal density [15-17] or functionality [18]. In this work, we compare the readability of patterns across different combinations of imaging and computing devices, for patterns on flat and non-flat surfaces. Because hue measurements are robust to different imaging environments, the progressive layer of the selected marking scheme is well suited for this use case. Still, it is important to verify that the proposed system can successfully image codes on different surfaces. As a mechanism to isolate the differences between the color layer and the traditional layer of a Data Matrix code, a first test compares reading capabilities on non-planar surfaces to help establish the effective information density of the different layers on non-planar surfaces. A goal of this evaluation is to stress-test the components involved in progressive barcodes.

The surface of a cylindrical 1.5 inches in diameter was used for the initial evaluation, with target markings square with the edges of the cylinder, i.e., oriented at 0 degrees, and rotated by 45 degrees (see Figure 5). Targets were traditional black and white (B&W) Data Matrix codes and Data Matrix codes rendered with different hues (cyan, magenta and yellow); examples are illustrated in Figure 6. This test focuses on the ability of the system, on average, to disambiguate regions composed of luminance- and hue-based elements to form a Data Matrix code. Fives sizes of markings were produced, the smallest of which was 0.27 inches on a side. Larger size samples were created by re-sampling by a factor of 2-5 via pixel replication in both dimensions. The goal of the test was to determine which sizes of which type of code could be read by each imaging system, at each orientation. Images captured against a flat surface were used as a control. The test evaluated three imaging setups: (1) scanning with a notebook running a local web-server using a Samsung Galaxy S6 as a virtual PC camera, (2) scanning with the same phone in a standalone web-connected setting, and (3) scanning with an Apple iPhone SE in a standalone web-connected setting.

Figure 5. *Diagram of surface depicting orientations of labels evaluated in the test.*

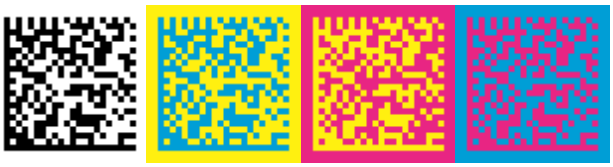

Figure 6. *Examples of traditional (BW) and hue-based (color) barcode markings tested for readability. The colors selected for the hue-based encodings isolate the ability distinguish between primary inks in an overprinting system.*

Tables 2 and 3 list readability results for sets of black & white (B&W) and color Data Matrix markings, respectively. Each table entry corresponds to how many reads succeed out of 3 different prints. In general, the PC + peripheral setup provided slightly better performance with the same device, except on small color marks. B&W marks were easier across the board, and all devices had difficult with larger marks rotated by 45 degrees. Failures occurred with examples that were both too large and too small (see Figures 7 and 8, respectively). In the case of the PC + phone tests, the imagery analyzed by the PC was necessarily compressed, which reduces the detail available in the images of the code. In all tests with codes larger than one square inch, the curvature of the surface impeded accurate decoding. The iPhone demonstrated the weakest performance on the color images, possibly due its limited resolution.

Table 2. B&W reads: imaging setup/view v. size multiplier.

Imaging setup/ view	x1 size	x2 size	x3 size	x4 size	x5 size
PC+phone/flat	all	all	all	all	all
PC+phone/0°	all	all	all	all	all
PC+phone/45°	all	all	all	-	-
Galaxy S6/flat	all	all	all	all	all
Galaxy S6/0°	all	all	all	all	all
Galaxy S6/45°	2/3	all	1/3	-	-
iPhone SE/flat	all	all	all	all	all
iPhone SE/0°	all	all	all	all	all
iPhone SE/45°	all	all	all	-	-

Table 3. Color reads: imaging setup/view v. size multiplier.

Imaging setup/ view	x1 size	x2 size	x3 size	x4 size	x5 size
PC+phone/flat	1/3	all	all	all	all
PC+phone/0°	-	all	all	all	all
PC+phone/45°	-	2/3	all	-	-
Galaxy S6/flat	all	all	all	all	all
Galaxy S6/0°	2/3	all	all	all	all
Galaxy S6/45°	1/3	all	2/3	-	-
iPhone SE/flat	2/3	all	all	all	all
iPhone SE/0°	2/3	all	2/3	2/3	2/3
iPhone SE/45°	1/3	2/3	1/3	-	-

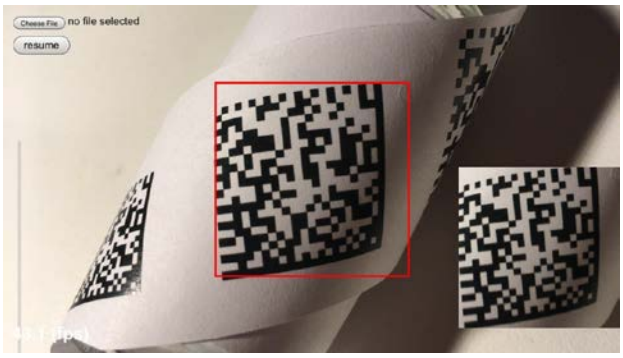

Figure 7. *Example failure case from an iPhone SE. The marking depicted in the image is a little larger than 1 inch on each side, wrapped around a cylinder 1.5 inches in diameter. Non-planar distortions are clear, since the marking, end-to-end covers a significant portion of the perimeter of the cylinder.*

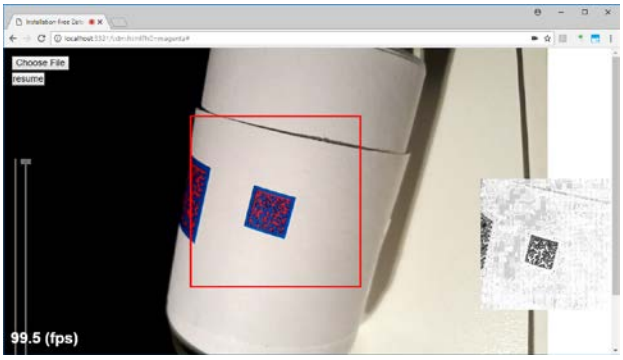

Figure 8. *Example failure case running on a PC using an Android phone as a camera (Samsung Galaxy S6) as a PC peripheral. Since the video stream from the camera is compressed, the resulting image is harder to decode.*

5. Conclusion

This work investigated a framework for tracking items through a production lifecycle, enhanced by the ability to track partially manufactured parts, using a single implementation capable of driving a range of imaging devices. The ability to track the suggested markings on non-planar surfaces was evaluated by scanning codes wrapped onto a cylindrical surface at two different angles. Whereas larger barcodes are generally easier to interpret, when rendered on top of highly non-planar objects, the ability to resolve the codes deteriorated; when the test markings were rotated by 45 degrees, end-to-end, the opposite corners extended nearly halfway around the cylindrical surface. Nonetheless, it was found that most codes (both color and B&W) roughly between ½ and ¾ of an inch on a side could be resolved and decoded.

Because of the rapid advances in web-accessible APIs, mobile camera capabilities and 3D color printing, proposed approaches will improve over time, and will also become easier to use. Even in the failure cases, the quality of the imagery obtained through the proposed software solution was quite high, and consequently, further enhancements should help improve decoding performance Next steps will require analyses using less regular non-planar surfaces, and exploration of web-accessible GPU capabilities to attain more computational efficiency.

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] International Standard ISO/IEC 18004:2015, Third edition. “Information technology – Automatic identification and data capture techniques – QR Code bar code symbology

[3] S. Simske and AM. Vans, “Applications for Progressive Barcodes,” J. Imaging Science and Technology, 58(4), 2014, pp. 40404-1-40404-9.

[4] M. Vans, S J. Simske, and B Loucks. “Progressive Barcodes,” Proceedings of the Digital Fabrication and Digital Printing Conference, NIP28, Quebec City, Quebec, Canada, 2012. pp. 368-370.

[5] MediaDevices.getUserMedia() – Web APIs | MDN, https://developer.mozilla.org/enUS/docs/Web/API/MediaDevices/getUserMedia, August 28, 2017.

[6] Amazon Rekognition, https://aws.amazon.com/rekognition/.

[7] Cloud Vision API, https://cloud.google.com/vision/.

[8] A. Zakai, Emscripten: An LLVM to JavaScript Compiler, https://github.com/kripken/emscripten, Dec. 23, 2016.

[9] Compiling a New C/C++ Module to WebAssembly, https://developer.mozilla.org/en-US/docs/WebAssembly/C_to_wasm, June 18, 2018.

[10] GPU Accelerated Javascript, https://github.com/gpujs/gpu.js.

[11] glfx An image effects library for Javascript,; http://evanw.github.io/glfx.js/, 2011.

[12] M. Vans, M. Gaubatz and S. J. Simske. “Embedding a Standard within a Standard using Mobile Progressive Barcodes,” Proceedings of Printing for Fabrication Conference, Denver, CO, 2017, pp. 13-16.

[13] DroidCamX Wireless Webcam Pro, https://play.google.com/store/apps/details?id=com.dev47apps.droidcamx&hl=en_US, April 10, 2018.

[14] iVCam - HD Webcam for PC, https://itunes.apple.com/us/app/ivcam-hd-webcam-for-pc/id1164464478?mt=8.

[15] M. Melgar, M. Farias, F. Vidal, A. Zaghetto, "A High-Density Colored 2D-Barcode: CQR Code-9", Proceedings of the 29th SIBGRAPI, October 2017.

[16] S. Lyons and F. R. Kschischang, "Two-dimensional barcodes for mobile phones," Proceedings of the 25th Biennial Symposium on Communications, May 2010.

[17] S. Simske, M. Vans and G. Adams, “Error-Correcting Code (ECC) and Module Size Considerations in 2D Aztec Barcode Readability”, J. Imaging Science and Technology, 54(6), November 2010.

[18] H. Wakaumi, "A High-Density ternary barcode detection system employing an envelope-differential composite method," Proceedings of IEEE SENSORS 2008, October 2008.

Author Biography

Matthew Gaubatz is a research scientist at HP Labs in Seattle, Washington. His main interests are mathematics, security printing and extracting information from N-D surfaces. He has a Ph.D. in electrical engineering from Cornell University.