https://doi.org/10.2352/EI.2025.37.10.IPAS-229

Improving Version Detection for JAB Codes

Simon Bugert, Marco Frühwein, Waldemar Berchtold, Ann-Katrin Riedel, Julian Heeger; Fraunhofer Institute for Secure Information Technology SIT / ATHENE; Darmstadt, Hesse/Germany

Abstract

The focus of the work is to improve the reading performance of JAB Codes. JAB Code is a polychrome barcode that is ISO standardized. The weakness of the standardized decoding algorithm is the very low reading performance of under 10% for very large and rectangular codes. In many IT security applications, however, large and rectangular codes are required for the huge payload. In this paper, we present three different methods to improve the decoder. These methods aim at determining the version size of the JAB Code to be read. This is the step after the JAB Code has been located by the finder patterns and before the decoding can take place. The three methods have their advantages and disadvantages in their accuracy and performance.

The evaluation compares detection rates and error performance for Baseline, Segmentation, Zero Crossing, and Local Sampling methods. The results show that Local Sampling achieves the highest detection rates, with 285 partial and 131 complete detections, while also maintaining the lowest error levels. The other methods perform significantly worse. The findings highlight that Local Sampling offers the best performance, effectively addressing the challenges of version size determination with improved accuracy and reliability.

Motivation

Barcodes are widely used across various industries and by consumers for several reasons. Barcodes improve efficiency by enabling fast and accurate data entry, reducing manual errors. They are cost-effective to implement and help businesses manage inventory and assets in real time. Barcodes also enhance security by tracking products throughout the supply chain. Their versatility allows them to be used across various industries for tasks such as automating processes and ensuring compliance. Overall, they streamline operations, increase accuracy, and support business scalability. The most popular barcodes today are EAN, QR code and DataMatrix code. In recent years, IT security applications with high payload requirements have emerged. One example is securing sovereign documents. Here, it is necessary to store the data together with its digital signature in a barcode. Unfortunately, these codes do not provide enough capacity. JAB Code [1] has been developed for these applications. It uses 8 colors and can carry about three times as much data as a DataMatrix code or a QR code. JAB Code is standardized by ISO/IEC [2]. A reference decoding algorithm is proposed in the standard. The quality assessment solutions for JAB Codes must be evaluated using the reference decoding algorithm. Reading devices are not required to use the standardized algorithm. There is an open- source implementation of the reference decoder. The implementation shows a very good reading performance for small and medium JAB Codes. Large JAB Codes are not decoded reliably, which is less than 10%. This work focuses on optimizing the reference decoding algorithm by improving the detection of the size version.

First, we give an overview of the relevant part of the reference decoding algorithm. The reference algorithm must be used in the quality evaluation of the JAB Code. With the specified reference decoding algorithm specified in ISO/IEC 23634, almost any printed JAB Code of version size 20-32 would fail the quality assessment. At the same time, a company commissioned to print JAB Codes on documents must comply with a minimum quality level. In the first step, the decoding algorithm searches for the finder patterns using line scanning. A detailed description of the finder pattern search is given in this paper. It is important that the module size is also determined during the search using the finder patterns. The module size is then used in the next step to determine the number of modules in the width and the height of the code. To sample the modules in the next step, the distance between the finder patterns in x- and y- direction is simply used and divided by the size of the modules. The reader will immediately notice that a small error for the module size increases proportionally with the size of the version, i.e. the more modules the code has between two finder patterns. This is the reason for the weakness with large square or rectangular codes. To compensate for this error, the JAB Code has so-called alignment patterns. These can be found at different places in the code, depending on the code version. However, the alignment patterns cannot be reliably found if an incorrect version size was calculated in the previous step. Therefore, the detection depends on the correct determination of the version size. In the following, we would therefore like to present three different methods for determining the version size of the JAB Code to be read.

Related work

JAB Code follows the ISO/IEC 23634 standard [2]. An open-source implementation, along with an Android application, can be found in a GitHub repository[1]. Additionally, a demo with the ability to create and scan codes is accessible on the product website[2]. Several studies have explored applications of JAB Code in the field of IT security. In [5], the authors introduce a method for digitally signing documents that remain verifiable even after printing. [3] presents a ready-to-use solution for digitally signing various types of ad hoc documents. This solution is implemented as a progressive web app, emphasizing high security and privacy standards. In [6], the color mapping for JAB Codes by introducing a color calibration procedure. These works highlight the challenges associated with ensuring the reliable readability of JAB Codes. This work aims to further improve the performance of the JAB Code reader by optimizing version size detection.

[1] https://github.com/jabcode/jabcode
[2] https://www.jabcode.org

State of the art

The current method for JAB Code detection and decoding relies on the standardized algorithm. A key aspect of this algorithm is the calculation of module size during line scanning of each finder pattern, which is determined based on a five-module pattern. The code version is calculated based on the module size and the pairwise distance between the finder patterns. This approach is fundamental for identifying and decoding the code accurately because the location of each sampling point is based on these calculations. However, scaling issues arise when dealing with larger codes. Minor inaccuracies and rounding errors in module size estimation can accumulate, leading to errors that affect the decoding process. Specifically, these small errors may result in selecting the wrong side version of the JAB Code and thus choosing the wrong number of sampling points and thereby compromising readability and data integrity.

Approach

Preprocessing: Bresenham's algorithm

To extract the pixel values along the line inbetween two corner points, Bresenham's algorithm is used. Each of the following approaches uses the Bresenham's algorithm.

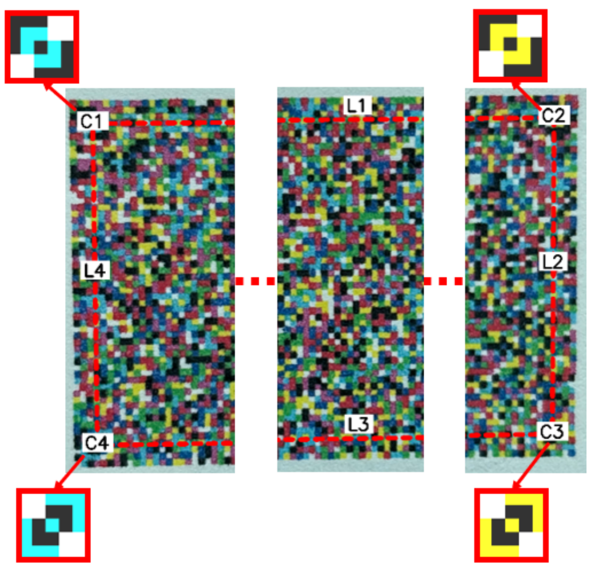

Figure 1: Corner points (C1-4) and Lines in- between (L1-L4).

Approach 1: Segmentation through color value assignment

This approach utilizes the expected color values of the JAB Code by first processing the individual color channels with a variance-based window and then mapping the resulting values accordingly to obtain binarized color profiles. The procedure is structured as follows:

1. Splitting up the RGB image into separate channels
2. 3x3 local variance filtering
3. Storing values into channel-specific profiles
4. Filtering the profile with a 1x9 mean window and using values as a threshold
5. Apply threshold according to the value at the current position

$$f(x) = \begin{cases} 255, & \text{if value} > \text{mean} \\ 0, & \text{otherwise} \end{cases}$$

6. Color assignment of the channels value combination

Since only 8 distinct color values are expected in the profile, as the JAB Code consists of just 8 colors, the three variance profiles from the RGB channels are then mapped to the 8 possible colors based on the current binarized value at each position across the three profiles (table 1).

Table 1: Color assignment values

	Red	Green	Blue
Red	255	0	0
Green	0	255	0
Blue	0	0	255
Yellow	255	255	0
Magenta	255	0	255
Cyan	0	255	255
Black	0	0	0
White	255	255	255

After going through the individual steps, the middle position of equal values is then used as the representative center of the module and is shown with a white line in figure 2. A correct color assignment is not important, only the recognition of a color change, as only the number of centers is counted for the calculation of the version size.

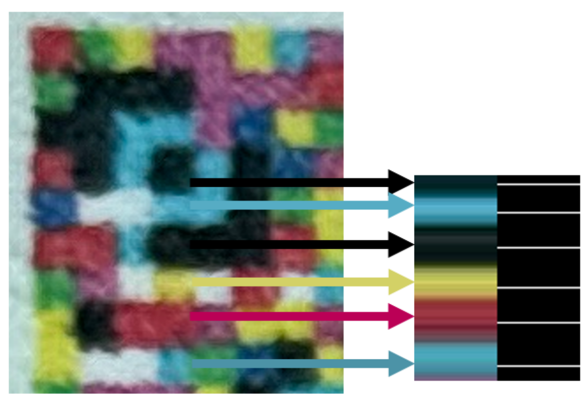

Figure 2: Cropped image on the right side next to the widened profile with the position of the module center (white line) found next to the original image to illustrate the validation.

Approach 2: Hue derivative zero crossings

This method takes advantage of the fact that a maximum/minimum color value (hue channel) is located in the middle of a module and changes color towards the edge of the module. As a result, maxima are the central midpoints, whereby further smaller secondary maxima are created that could be incorrectly recognized. For this reason, the profile is filtered further and the first derivative is formed so that the zero crossings can be assigned to clear maxima. The Hue channel (figure 3) is particularly suitable for this, as a single color channel contains the information about the color tones. The procedure is as follows:

1. Convertion of RGB image to HSV
2. Apply 7x7 median filter to Hue channel
3. Calculating derivative of values
4. Using zero-crossing as indicator of significant changes, representing center position of a module (figure 4)
5. using distance between two zero crossings as estimation of single module size

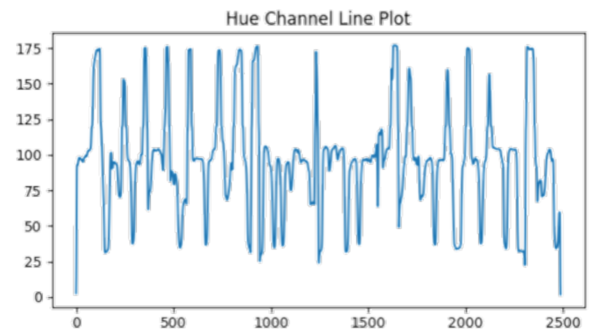

Figure 3: Hue channel after smoothing operations.

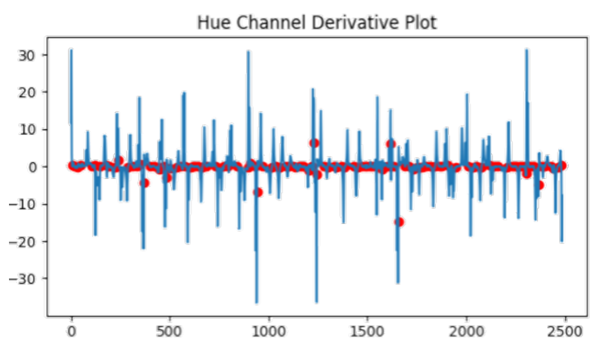

Figure 4: First derivative of the hue channel. Zero crossings are marked red.

Approach 3: Local Sampling

The third approach is based on the idea that the calculated module size at the FPs is approximately right and that the error adds up on a longer distance. By trying to correct the mistake for every step to the next module, the algorithm keeps the added errors low enough to place each sampling point in the modules. As with the other approaches, Bresenham's line algorithm is used to determine the connecting line between the finder patterns FP_a and FP_b. The *local module size* (*LMS*) refers to the estimated module size at a point between FP_a and FP_b, calculated through linear interpolation between the module sizes at these two points. The approach follows the following steps:

1. Start at FP_a
2. Repeat until the last finder pattern (FP_b) is reached within a small radius:
   (a) Place the next sampling point *LMS* pixels away from the last point in the direction of FP_b.
   (b) Optimize the sampling point:
      i. Compute the mean value in a neighborhood of size $r/4$ around the point.
      ii. Move the sampling point in both directions on the line using the maximum distance and calculate the mean within the window.
      iii. Minimize the RMS error between the original center and moved positions.
      iv. Position the point where the error is minimized.

Evaluation

Approach 1: Segmentation through color value assignment

Approach 1 shows that larger modules are not recognized with this method and therefore the module width deteriorates in the calculation. In the case of levels, several color-near positions are detected in a module, e.g. red and magenta or blue and cyan. This is caused by slight color fluctuations within the color profile. These can occur more frequently due to printing properties, paper properties or exposure. The advantage of this method is that it requires no prior knowledge and provides a good estimate of the approximate module size using pure image processing methods.

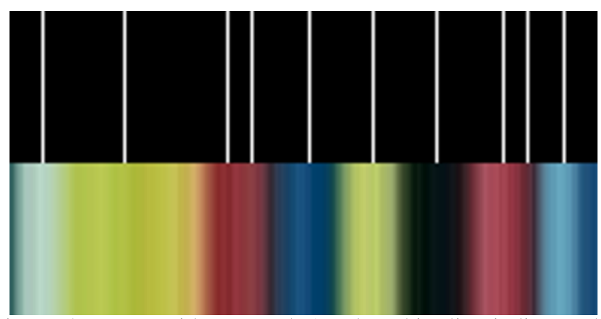
Figure 5: Issues with approach 1: The white line indicates the detected center of each module. Larger modules (e.g., the yellow module) are mistaken as a single unit, while variations in red and magenta cause multiple detections within a single module.

Approach 2: Hue derivative zero crossings

Approach 2, like approach 1, also has problems with larger modules. Possible centers of modules are detected at the junctions of two modules, as the color value also takes on an extreme value here.

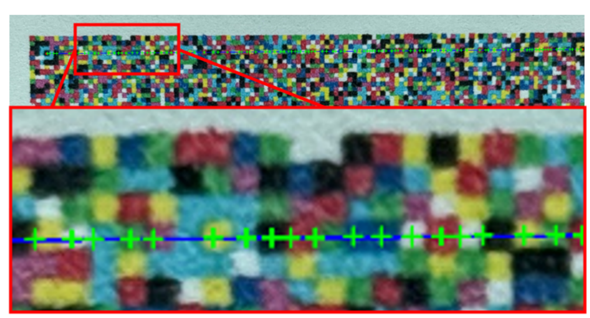
Figure 6: Issues with approach 2: Some modules are not detected and single modules cause multiple detections.

Approach 3: Local Sampling

This visualization highlights how the optimization adjusts point positions to improve alignment and accuracy, while some points remain unchanged due to their already optimal placement. In general, the points are optimized towards the center of the modules. The test shows initial positive results and demonstrates the feasibility of the method.

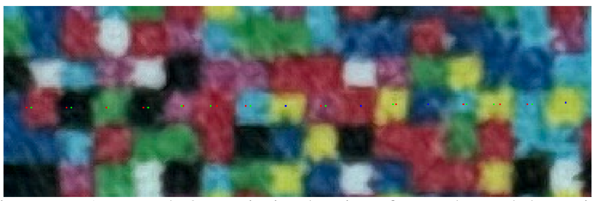

Figure 7: Approach 3: optimized points for each module: original points (red), optimized points (green) and unchanged points (blue)

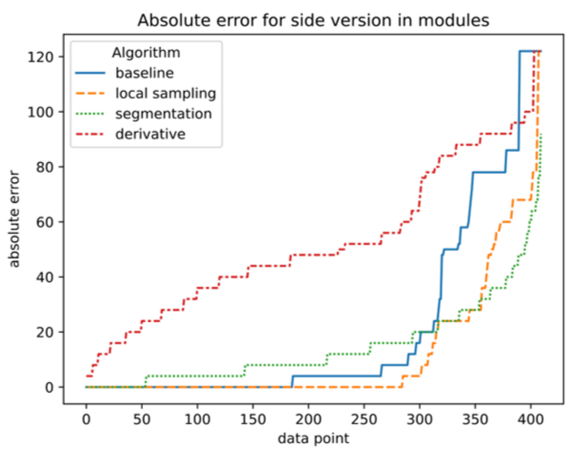

Figure 8: Absolute error for side version for baseline and three new algorithms.

Test Setup

Consisting of 5691 images, the data collection is made up of images from various smartphones of JAB Codes on security paper, printed with various printers. 564 images were selected from this test set and labeled by hand. The paper properties, especially existing prints in relation to the security properties, and print properties in combination with smartphone images represent a challenge (figure 9), but a link to the real application.

Results

Table 2: Detection results

Method	Complete	Partial (vertical or horizontal)
Baseline	67	186
Segmentation	7	54
Zero-Crossings	0	0
Local sampling	131	285

This section discusses the performance of different methods evaluated. For a subset of 205 images, the finder patterns could be detected. This results in two data points (horizontal and vertical) per image. The comparison focuses on partial and complete detections across the four presented methods: Baseline, Segmentation, Zero Crossing, and Local Sampling. Table 2 summarizes the number of partial and complete detections for each method. Local Sampling outperforms other methods with 285 partial and 131 complete detections, demonstrating superior detection capability. In contrast, the Baseline method achieves 186 partial and 67 complete detections, while Segmentation produces significantly lower results (54 partial, 7 complete). Zero Crossing fails to detect any points.

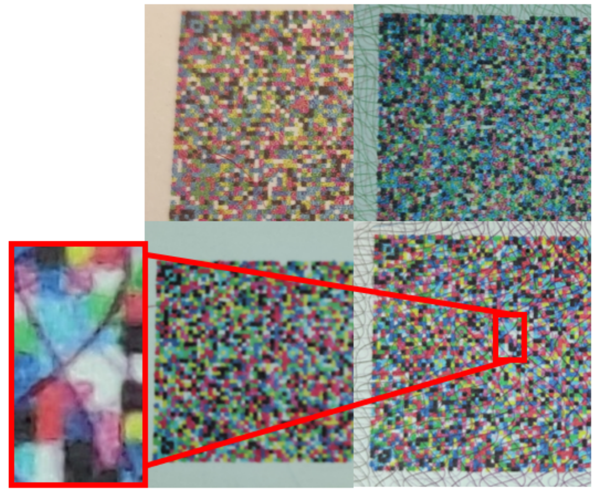

Figure 9: Test setup images. The color of the paper, printer differences and color structures present a major challenge.

The error distribution plot in figure 8 highlights the absolute error for side versions in modules across the four algorithms. The Local Sampling method (orange dashed line) consistently shows lower error values compared to the baseline (blue line), Segmentation (green dotted line), and Derivative (red dashed line). This suggests that Local Sampling not only achieves higher detection rates but also maintains better accuracy.

Discussion

In this work, three different approaches are used to try to solve the decoding problem of JAB Code that occurs with large version sizes. The version size of the captured JAB Code must be determined directly after the finder patterns have been found. To do so, it is necessary to determine the number of modules between the finder patterns. The challenges for determining the number of modules are manifold in practice. The module sizes vary, the colors and color transitions of the modules are distorted twice, namely during printing and then during capturing process. Colors may not change between adjacent modules. Geometric and linear distortions can occur. The module sizes vary arbitrarily from print to print and image to image. The complexity of a method for determining the JAB Code version must be low to ensure fast decoding. The three methods described try to reconcile these different challenges and show good initial results.

Overall, Local Sampling demonstrates the best performance, combining the highest detection rate with relatively low error values. This method proves to be the most reliable among those evaluated.

Conclusion and future work

In this work, we have investigated different approaches to improve version detection in JAB Codes, addressing a key limitation of the standardized decoding algorithm. Our analysis demonstrated that inaccuracies in module size estimation lead to significant decoding errors, particularly for large and rectangular JAB Codes.

We introduced and evaluated three methods — Segmentation, Zero Crossing, and Local Sampling — for determining the version size after finder pattern detection. Among these, the Local Sampling method showed the most promising results, achieving the highest detection rates while maintaining the lowest error levels.

While local sampling has shown improved performance, further refinements could increase its accuracy and reduce computational complexity while maintaining accuracy. Improving robustness against varying lighting conditions, different print qualities, and distortions could make the method more reliable in practical scenarios. In addition, parameter fine-tuning and improved image pre-processing could further optimize detection performance. Future research should also explore machine learning approaches as an alternative to standard methods, potentially leading to more adaptive and accurate version size determination.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research (BMBF) and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

[1] Berchtold Waldemar, Liu Huajian, Steinebach Martin, Klein Dominik, Senger Tobias, Thenee Nicolas. (2020). JAB Code - A Versatile Polychrome 2D Barcode. Electronic Imaging. 2020. 207-1. 10.2352/ISSN.2470-1173.2020.3.MOBMU-207.

[2] ISO/IEC 23634:2022 (2022). Information technology — Automatic identification and data capture techniques — JAB Code polychrome bar code symbology specification. Standard, International Organization for Standardization, Geneva, CH.

[3] Bugert Simon, Heeger Julian, Berchtold Waldemar. (2023). Integrity and authenticity verification of printed documents by smartphones. Electronic Imaging. 2023. 10.2352/EI.2023.35.3.MOBMU-352.

[4] Waldemar Berchtold, Dani El-soufi, Martin Steinebach, ”Smartphone-supported integrity verification of printed documents” in Proc. IST Int’l. Symp. on Electronic Imaging: Media Watermarking, Security, and Forensics, 2022, pp 325-1 - 325-5, https://doi.org/10.2352/EI.2022.34.4.MWSF-325.

[5] C. Winter, W. Berchtold, J. N. Hollenbeck, Securing Physical Documents with Digital Signatures, In: 2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP). Vol. 09. 2019, pp. 1–6.

[6] Bugert Simon, Heeger Julian, Berchtold Waldemar. (2024). Color Calibration for Multicolored Barcodes Using Smartphones. Electronic Imaging. 36. 312-1. 10.2352/EI.2024.36.3.MOBMU-312.

Author Biography

Simon Bugert received his Master’s degree in computer science from the Technical University of Darmstadt, Germany in 2021. Since then, has been a research associate in the Media Security and IT Forensics department at the Fraunhofer Institute for Secure Information Technology (SIT) and at the ATHENE National Research Center for Applied Cybersecurity.

Marco Frühwein has been a research associate at Fraunhofer SIT since 2021. He obtained his Bachelor’s degree in 2017 and subsequently his Master’s degree in 2018 at Darmstadt University of Applied Sciences in optics and computer vision. During his studies, he focused on digital image processing. He is working on projects for forgery-proof barcode systems and deep learning-based recognition of archaeological objects from camera images. He is also involved in the detection of AI-based content and has developed his own diagnostic tools for this purpose. He regularly works on IT forensic reports for the forensic evaluation and authentication of image and video data.

Waldemar Berchtold has headed the Multimedia Security research group since 2022 at the Fraunhofer Institute for Secure Information Technology (SIT) and is a researcher at the National Research Center for Applied Cybersecurity (ATHENE) in Darmstadt, Germany. He received his diploma in mathematics in 2008 and his Ph.D. in 2022 at TU Darmstadt. The focus of his research is in various areas of multimedia security for authenticity and integrity proof and digital watermarking. He has led numerous projects in the field of media security with a focus on audio and video.

Ann-Katrin Riedel has been a research associate at Fraunhofer SIT since 2019. She received her bachelor’s degree in 2017 and her master’s degree in 2019 at Darmstadt University of Applied Sciences in the field of optics and computer vision. She focused on digital image processing during her studies. She worked on image-based robust machine learning models and algorithms. Later on, she worked on AI-based content related to images and on projects for counterfeit-proof barcode systems. In addition, she occasionally works on IT forensic reports for image and video forensic evaluation and authentication.

Julian Heeger is a research associate in the Media Security and IT Forensics department at the Fraunhofer Institute for Secure Information Technology (SIT) and a researcher at the National Research Center for Applied Cybersecurity (ATHENE) in Darmstadt, Germany. He holds a Master’s degree in IT security from the Technical University of Darmstadt.