Blind Print-Cam Data Hiding Exploiting Color Perception

Federico Baldessari[1], Giulia Boato[1,2][0000−0002−0260−9528], and Federica Lago[1,2][0000−0001−7840−7693]

[1] University of Trento, Trento TN 38123, Italy
federico.baldessari@alumni.unitn.it
[2] {giulia.boato, federica.lago-1}@unitn.it

Abstract. Augmented Reality is becoming a fundamental technique to provide an easy access to additional information directly from the surrounding environment. It is however crucial that the mean through which the information is accessed is as integrated in the environment as possible. To this end, several data hiding techniques have been devised in the years to encode information in images in an imperceptible way. However, these techniques are frequently strongly affected by printing and re-acquisition process. This work presents an application developing a data hiding technique robust to printing and camera acquisition (print-cam), thus allowing to recover inserted data (hundreds or thousands of information bits) from printed images in a robust way (e.g., for different size of printed cover image, in different illumination conditions, with various geometric distortions). Performance and robustness of the proposed solution are tested with respect to different metrics to prove the feasibility of the technique.

Keywords: Data Hiding · Augmented Reality · Print-cam process

1 Introduction

The fusion between the digital world and the real world is becoming possible thanks to advances in Augmented Reality (AR), that allows digital information to be shown to a user in addition to the surrounding real-world environment. In this way, it is possible to alter the perception of the environment without leaving the physical world, giving the user a more realistic experience than in Virtual Reality where the environment is entirely replaced by a simulated one. To create an immersive experience, a clear understanding of the world is essential. Thus, several solutions based on image recognition were devised, thanks to the ability of this technique to identify different objects and scenarios. However, the insertion of new data can be challenging, and a server is required to store the information associated to objects. Thus, QR-Codes and other similar markers are frequently adopted thanks to their convenience and to the amount of data they are able to store, although they are not appealing.

In order to develop our application, we decided to focus on data hiding techniques that grand an imperceptible and robust information embedding [2–4]. The

most common techniques are those based on transform domain watermarking, that, however, also present some drawbacks: Discrete Cosine Transform (DCT) methods [2, 14] are not able to work properly under strong geometric distortions, while Discrete Wavelet Transform (DWT) [1, 11, 15] might lead to low performances due to small shift that might occur when reading the print-cam version of an image. Furthermore, due to the lack of directional selectivity the watermarking capacity of DWT methods is small [8].

A work similar to the one we propose is the one in [1], where they devised a watermarking method suitable for mobile applications. However, this work does not take into account distortions or print-cam operations that are fundamental for real scenarios. Indeed, the process of printing an image and then taking a photo of the printed picture to get back its digital representation, makes the final image different from the original one due to several random distortions, including global geometric distorts, local random nonlinear geometric distortions, and non linear pixel value distortion. This makes the context very different from classical watermarking techniques which consider many different types of robustness attacks but rarely this specific print-cam process.

Different studies dealing with print-scan have been published, although there is no ideal method capable of resisting print-scan geometric problem (rotation, translation, and scaling) [5]. Print-cam is even more complicated since it introduces perspective distortion due to the position and angle of the camera with respect to the object, and illumination distortions. In 2006, Kim at al. [7] repeatedly embedded a 63-bit fingerprint in the spatial domain of color images printed re-captured images, so that the data could be extracted with auto-correlation. Unfortunately, this method required lots of manual work before being able to recover the hidden data. In 2008, Pramila et al. [15] proposed to apply conventional watermarking techniques in the print-cam process by introducing a frame around the image to remove geometrical distortion after extracting the corner points. However, the discrete wavelet transform based watermarking technique required to capture the image carefully since it was able to resist only light geometric distortions. In 2012, Thongkor and Amornraksa [16] proposed a print-cam watermarking scheme for photo authentication in Thai national ID card. The method was a spatial domain watermarking performed in the blue color channel of the picture. Although the method showed interesting results, the decoding was non-blind, requiring the original cover image. In 2017, Nam-Tuan Le [10] proposed spatial data hiding for screen monitor working on variations in the blue channel of the same image over time, that, however, allowed for the embedding of only 8-bit of information for each image. In 2018, Gourrame et al. [5] designed a zero-bit watermarking based on the Fourier transform and a specific correction pre-process. However, it was a zero-bit watermarking, meaning that the software was capable of detecting the presence or the absence of the watermark in the marked object but no information was extracted. Nguyen at al. [12] in 2017 presented another interesting approach where the image is divided into hundreds of small tiles and the gaps between the tiles are used to encode black or white regions of a given QR-Code. While the modification to the original image

remains highly impacting and the decoding phase showed some problems, the idea to use QR-Code as a wrapper for the secret message remains interesting. To the best of our knowledge there are no methodologies allowing for high capacity, not strongly visible impact, blind and robust detection.

Here we devise a solution that combines all advantages mentioned above: it is a blind data hiding technique, imperceptibly encoding a large amount information (up to 4096 bits) inside an image in such a way that the receiver can retrieve the full information without prior knowledge on the original image, and it is robust against printing and re-capturing, allowing anyone to use a standard printer for inserting new information in the digital world that can be simply captured with a smartphone or a tablet to have access to advertisement or AR applications.

2 Methodology

In this paper, we devised a method to encode and decode hundreds or thousands of bits in or from an image in a blind and fully automatic way, requiring robustness to print-cam process. The various steps, discussed in the following sections, are combined into a real-time application[3] accessible from smartphones, tablets and computers with different operating systems, which allowed us to extensively test the feasibility of the devised solution for several devices and real scenarios.

Encoding This procedure works block-wise to embed a message $\mathcal{M}$, constituted by an $M \times M$ matrix of zeros and ones, into an image $\mathcal{I}$ of size $N \times N$, with $N > M$. Thus, each bit in $\mathcal{M}$ is encoded in a square of size $K \times K$, with $K = N/M$. This is done by creating an *activation matrix* AM of size $N \times N$, composed by $M \times M$ squares of size $K \times K$ encoding a vertical line if the bit to encode is a 0, and a horizontal line otherwise. For the embedding blue channel was chosen as changes this channel affect less the image quality [2, 9, 13] and because human eyes are less sensitive to blue variations [6]. For each pixel in the blue channel ($c = 2$) of $\mathcal{I}$ in position $(x,\ y)$, the embedding follows Eq. (1)

$$\mathcal{I}^*(x, y, c) = \mu_b + AM(x, y) \cdot AV \tag{1}$$

where $\mathcal{I}^*$ is the encoded image, and μ_b is the average in the blue channel of the $K \times K$ square in which the pixel (x, y) is contained. Bits in $AM(x, y)$ are scaled by an *activation value* AV that can either be a constant or work as dynamic masking following Eq. (2)

$$AV = 50 \cdot \left(\frac{\sigma_r^2 + \sigma_g^2}{2 \cdot 255} + 1 \right) \tag{2}$$

where σ_r^2 and σ_g^2 represents the variances of the red and green channels. As depicted in Fig. 1, using Eq. (2) allows to better exploit human color perception to

[3] https://wizardly-bardeen-e48e37.netlify.com/

achieve invisibility while making the encoded message robust. Given the advantages of the dynamic masking, experiments in Section 3 were performed using it. Finally, the image is normalized to fit the range [0-255] and a black border of fixed size is added to ease the detection.

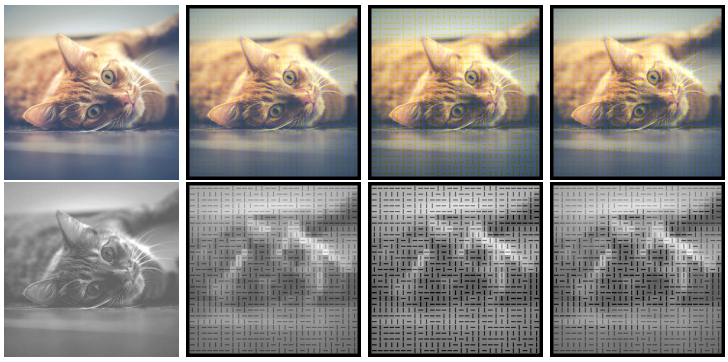

Fig. 1. From left to right the cover image and the images encoding the same message with $AV = 50$, $AV = 100$ and dynamic masking with $AV \in [50 - 100]$ respectively. The top row shows the color image, while the bottom row the blue-filtered images.

Decoding The decoding algorithm is able to retrieve the embedded message from a printed encoded image, robustly with respect to size, position and rotation. The algorithm has two phases: image identification and message decoding. The first phase consists in identifying with standard OpenCV functions the biggest square with a black border, which is removed. Then, some geometrical distortions are corrected using Four Point Perspective Transformation.

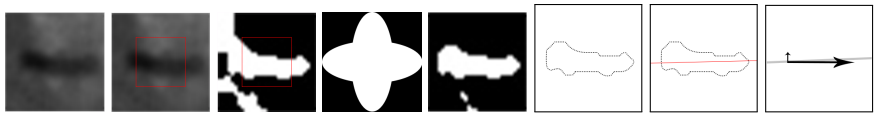

Fig. 2. Steps performed to identify and extract a line and determine its orientation.

The decoding phase blurs the image identified and normalized in the previous phase to decrease noise. Then it divides the image into squares and the message is decoded following the steps depicted in Fig. 2: i) identifying a safe region as an inner square of size 50% of the square size, ii) detecting the contours inside the region that might identify a line, iii) using a mask composed by two perpendicular ellipses to remove false-positives, iv) fitting a line inside the contours, and v) extracting the orientation of the line. The orientation is then used to decide whether the bit encoded was a zero (in case of a vertical line) or a one (in case of a horizontal line).

3 Experimental evaluation

The proposed method was tested in real condition of natural and artificial light. To do this six images have been printed on paper using a laser printer with a resolution of 600 ppi in different sizes. Of these, five encoding 29×29 messages (841 bits), five encoding 49×49 messages (2401 bits) and one encoding a 64×64 message (4096 bits). Examples on how the different encoding affect the quality of the image, can be seen in Fig. 3, where it also possible to notice an high invisibility of the message.

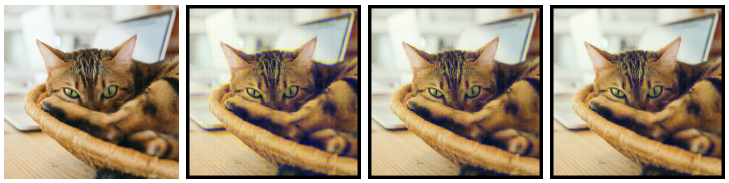

Fig. 3. From left to right the cover image, and the images with a *Lorem Ipsum* message encoded with 29×29, 49×49 and 64×64 bits.

Then, for each experiment, different pictures were captured freehandly using either a reflex camera (Nikon D3100), a smartphone (Xiaomi Mi 8 Lite) or a tablet (iPad Mini 4). Since the reflex does not support the application directly, the images captured by this device have been decoded using an Asus Zenbook. The pictures analyzed in the experiments on exposures and resolution were shot using the Nikon camera, since it allows a better understanding of the results thanks to a higher resolution that reduces the number of mistakes, as demonstrated in the last experiment.

Experiment on different exposures The goal of this experiment is to understand how random variations in pixels introduced by different light conditions affect the performances of the devised algorithm. Since digital cameras tend to mitigate the problem of low or high light by automatically adjusting the exposition time, we manually changed the exposition time while maintain the same light condition. Varying the exposition time, it was possible to shot 20 pictures (for a total of 140 images), using as target the using as target the 64×64 encoded image, capturing different levels of light as shown in Table 1. The darker the image the worst the results. In dark conditions, in fact, the errors are more than 8% on average with a peak at 9.32%. Another notable result is that overexposed images have better results than images with balanced light since the line seems to stand out with respect to the background. For -0.6 the average error was 0.02%, while for standard light condition it was 0.48%.

Table 1. Performance in terms of error rate for different exposures [-0.6 - 0.6]

	-0.6	**-0.4**	**-0.2**	**0**	**0.2**	**0.4**	**0.6**
Avg	0.02	0.04	0.17	0.48	1.34	3.52	8.11
Std	0.01	0.03	0.06	0.09	0.17	0.26	0.65

Experiment on rotation Perspective transformations lead to errors caused by the approximations performed by the algorithm to guess pixels that are not in the original data that might affect performances. A total of 320 shots of the 64×64 encoded image (40 for each degree of rotation) were taken at different angles, considering a rotation on the y-axis between 5 and 60 degrees. The error rate reported in Table 2 starts to strongly increase from 15 degrees, reaching a 27.31% for a rotation of 60 degrees. Nevertheless, the accuracy for lower rotations is sufficiently high to allow robustness and let the method be suitable for many real scenarios.

Table 2. Performances in terms of error rate for different degrees of rotation

	5°	**10°**	**15°**	**20°**	**30°**	**40°**	**50°**	**60°**
Avg	0.41	0.48	0.71	1.39	2.08	2.86	7.86	27.31
Std	0.16	0.12	0.16	0.22	0.52	0.46	1.09	4.34

Experiment on resolution Another important aspect to take into consideration is the distance from the target, as it affects the final resolution of the target image and hence performances. To test this, 50 pictures were taken at three different distances from the 64×64 encoded image. This translates to three different resolutions: high (2818×2815 pixels - full resolution), medium (1584×1581 pixels - about 30% of the original resolution), and low (1099×1096 pixels - 15% of the original resolution). As can be seen in Table 3, the performance decreases the more remote the camera is from the code of interest. However, even for the smallest resolution tested, the error remains rather low.

Table 3. Performances in terms of error rate for different resolutions.

	High	**Medium**	**Low**
Avg	0.13	0.66	5.39
Std	0.69	0.14	0.77

Experiments on device performances Different devices have been taken into account to prove that the results are not device-dependent. The performance of the devices was measured in terms of error rate and time to decode the message. For these tests 20 shots for each of the 5 sample images were taken for each tested device, for a total of 300 images. Results obtained for some sample images for the three different devices can be seen in Table 4, which proves that it is possible to encode 841 bits with error rate always below 1.3% even for the smallest images. The Nikon and the Xiaomi were capable of decoding also 2401 bits of information in different conditions. The iPad showed severe limits in decoding caused by a lower resolution compared to other devices under the same conditions (see Fig. 4).

Since we are targeting real-time applications, an important analysis concerns the time needed by the algorithm to decode the message. Table 5 shows the average response time for different cameras to decode 841 bits and 2401 bits information respectively. The time is reasonable for most scenarios, however, the Xiaomi appears to be the one with lower performances. This suggests that

our method works better for high-end devices and that future work should focus on optimizing the algorithm to work also on lower-end devices.

Table 4. Error rate on several sample images with different encoded messages and print size in cm (reported in the first and second rows respectively) for different devices.

	29 × 29 (841 bits)						49 × 49 (2401 bits)			
	13 × 13		6.5 × 6.5		3.25 × 3.25		13 × 13		6.5 × 6.5	
	Avg	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**
Nikon	0.00	0.00	0.00	0.00	0.06	0.07	0.07	0.03	0.39	0.40
Xiaomi	0.00	0.00	0.01	0.03	1.00	1.27	0.18	0.07	5.19	3.93
iPad	0.39	0.15	0.30	0.27	1.32	1.40	15.10	2.00	28.07	3.89

Table 5. Average decoding time (in seconds) for 29×29 and 49×49 messages

	Nikon	**Xiaomi**	**iPad**
29×29	0.12	0.98	0.40
49×49	0.24	2.9	0.92

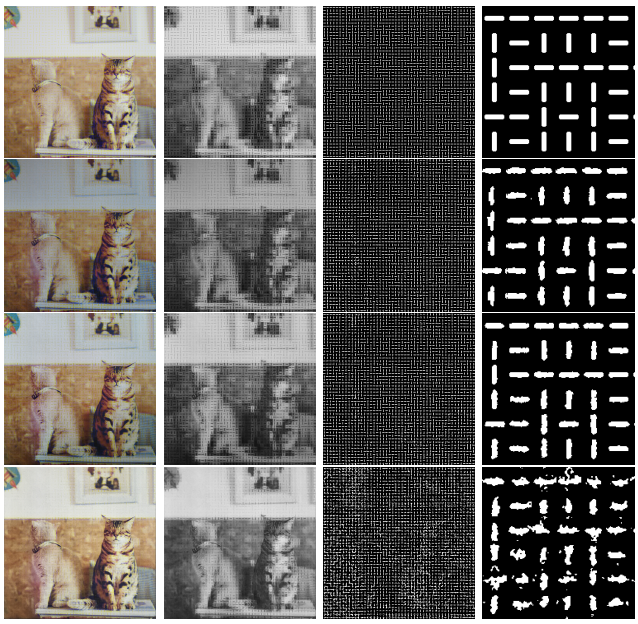

Fig. 4. From left to right the original image, the blue-filterd image, the decoded message and a 6×6 zoom into a portion of the message. From top to bottom the original image and the image captured with the Nikon, the Xiaomi and the iPad.

4 Conclusions

This paper describes an application developing a blind print-and-cam data hiding that allows encoding of large amounts of information and fast and robust detection, making it suitable for user-driven real-time applications. Performances are good for high-end devices, allowing room for improvements on old devices, where a QR-Code is a lot faster. The results show that the method is robust to light, rotation and resolution changes in the print-cam scenario. Moreover, it is possible to correctly read thousand of bits in good light condition, in particular for high-resolution devices which effectively decode four thousand bits.

References

1. Al-Otum, H., Al-Shalabi, N.E.: Copyright protection of color images for android-based smartphones using watermarking with quick-response code. Multimedia Tools and Applications **77**(12), 15625–15655 (2018)
2. Barni, M., Bartolini, F., Piva, A.: Multichannel watermarking of color images. IEEE Trans. on circuits and systems for video technology **12**(3) (2002)
3. Boato, G., et al.: Watermarking robustness evaluation based on perceptual quality via genetic algorithms. IEEE Trans. on Inf. Forensics and Security **4**(2) (2009)
4. Cancellaro, M., et al.: A commutative digital image watermarking and encryption method in the tree structured haar transform domain. Signal Processing: Image Communication **26**(1) (2011)
5. Gourrame, K., et al.: A zero-bit fourier image watermarking for print-cam process. Multimedia Tools and Applications **78**(2) (2019)
6. Hänninen, H., Nuutinen, M., Oittinen, P.: Visibility and annoyance of digital watermarks. Graphic Arts in Finland **36**, 1 (2007)
7. Kim, W.g., Lee, S.H., Seo, Y.s.: Image fingerprinting scheme for print-and-capture model. In: Pacific-Rim Conference on Multimedia. Springer (2006)
8. Kingsbury, N.: Complex wavelets for shift invariant analysis and filtering of signals. Applied and computational harmonic analysis **10**(3), 234–253 (2001)
9. Kutter, M., Jordan, F.D., Bossen, F.: Digital watermarking of color images using amplitude modulation. Journal of Electronic imaging **7**(2) (1998)
10. Le, N.T.: Invisible watermarking optical camera communication and compatibility issues of IEEE 802.15.7r1 specification. Optics Communications **390** (2017)
11. Najafi, E.: A robust embedding and blind extraction of image watermarking based on discrete wavelet transform. Mathematical Sciences **11**(4), 307–318 (2017)
12. Nguyen, M., et al.: A tile based colour picture with hidden qr code for augmented reality and beyond. ACM Symposium on Virtual Reality Soft. and Techn. (2017)
13. Parisis, A., Carré, P., Fernandez-Maloigne, C.: Colour watermarking: study of different representation spaces. In: Colour in Graphics, Imaging, and Vision (2002)
14. Poljicak, A., Mandic, L., Agic, D.: Discrete fourier transform-based watermarking method with an optimal implementation radius. Journal of Electronic Imaging **20**(3), 033008 (2011)
15. Pramila, A., Keskinarkaus, A., Seppänen, T.: Watermark robustness in the print-cam process. Signal processing, pattern recognition, and applications (2008)
16. Thongkor, K., Amornraksa, T.: Digital image watermarking for photo authentication in thai national id card. In: IEEE Conf. on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (2012)