Imperceptible Color Vibration for Screen–Camera Communication via 2D Binary Pattern

Satoshi Abe†, Takefumi Hiraki ††, Shogo Fukushima †, Takeshi Naemura (member)†

Abstract Communication between screens and cameras has attracted attention as a ubiquitous information source, motivated by the widespread use of smartphones and the increase of public information screens. The method that encodes data into visible patterns impairs the user’s visual experience. Previously, embedding matrix barcodes into images on displays by utilizing imperceptible color vibration was proposed. In this approach, the visual experience is maintained considering that barcodes are imperceptible, and it can be implemented on almost any display and camera. Herein, we describe a sophisticated modulation protocol and restoration procedure whereby device characteristics such as the display’s gamma feature and the smartphone’s rolling shutter are taken into consideration. Extensive experiments reveal the parameters for the modulation and that this system works under practical situations. In addition, scenarios of potential practical applications and a user study examining imperceptibility of barcodes and usability of the system are presented to illustrate the technological capabilities.

Key words: Screen–camera communication, imperceptible color vibration, ubiquitous computing.

1. Introduction

Graphic display devices, in addition to their role as passive displays, have been recognized for their value as data transmission tools, because they can facilitate the interaction between visual images and computing devices. Screen–camera communication is a promising solution for ubiquitous communications, now that most people carry smartphones, and now that displays for digital signage are increasingly installed in public places.

Compared to other wireless communication technologies, screen–camera communication offers some advantages. First, it requires no initialization such as Bluetooth pairing or Wi-Fi setup, and it does not require additional hardware such as RF tags in near-field communication. Screen–camera communication can easily receive data from a host device in an operationally intuitive manner. For instance, Yang et al.1 proposed device-to-device communication called EM-Comm, which uses electromagnetic noise as a carrier signal; however, users require a special wristband to use EM-comm. Second, screen–camera communica-

---

1. System operation of the method. Two color-vibrated images are displayed alternately. A smartphone camera extracts the embedded matrix barcode, while the human eye perceives only a normal image.
into visible patterns. By contrast, embedding invisible data into displayed content improves the user’s visual experience because it eliminates a code pattern that is meaningless to the user\textsuperscript{13(15)(16)(17)(18)(19)}\textsuperscript{10}. These technologies leverage high-speed luminance changes that are imperceptible to the human eye. The maximum frequency of luminance flicker perceptible to the human eye is called the critical fusion frequency (CFF) and is known to be about 60 Hz\textsuperscript{10}. We propose embedding matrix barcodes into images by utilizing imperceptible color vibration (Figure 1). Unlike luminance, the maximum frequency of perceptible chromatic flicker is called the chromatic CFF (CCFF) at approximately 25 Hz\textsuperscript{11}. Therefore, we can realize imperceptible data transmission at a refresh rate of only 60 Hz. We proposed this concept previously\textsuperscript{12}, where each pixel corresponds to a data point decoded by a color sensor system. Whereas we demonstrated the first prototype of our screen–camera system\textsuperscript{13} and implemented the near-screen interaction application with an array of imperceptible AR Makers based on this system\textsuperscript{14}, this paper further explores a sophisticated scheme of embedding the 2D binary pattern by using the imperceptible color vibration. In particular, we describe how the imperceptible code is embedded into images and extracted with a camera rather than introducing a method specialized for data transmission. Furthermore, we applied our color vibration method to a public display system, demonstrated three practical use cases, and showed the applicability of our method through a user study on imperceptibility of barcodes and usability of the system.

Our key contributions are summarized as follows:

• We designed the modulation protocol for embedding an imperceptible two-dimensional binary pattern into an image, along with the restoration procedure for extracting it from a screen with a camera.

• We realized screen–camera communication by embedding QR code that only requires an ordinary 60 Hz LCD display and an off-the-shelf smartphone.

• Experiments revealed the parameters for the modulation and demonstrated that this system works with general images on several displays and smartphones.

• We showed the technological capabilities by developing the applications in practical scenarios and examining imperceptibility of embedded barcodes and usability of the applications through a user study.

2. Related Work

2.1 Visible Light Communication

Visible Light Communication (VLC) is a communication scheme that utilizes visible light as a data source for devices as well as the light source for the user\textsuperscript{15}. Based on communication from an LED to a photo diode\textsuperscript{16}, VLC is applied to LED–camera communication\textsuperscript{17}, projector–photo sensor communication\textsuperscript{18(19)(20)}, and screen–camera communication (discussed in Section 2.2). The imperceptible color vibration used by the proposed method is also a kind of VLC. The feature of imperceptible color vibration is that it realizes invisibility by using high-speed chromatic flicker instead of high-speed luminance flicker.

2.2 Screen–Camera Communication

To realize screen–camera communication, researchers have encoded data into visible patterns with elaborate designs. The most common example of this technique is the QR code\textsuperscript{21}, which displays a black-and-white cell matrix barcode. Ample research has explored boosting the data capacity and transmission robustness. Pixnet\textsuperscript{22} is an orthogonal frequency division multiplexing (OFDM)-based matrix barcode scheme whereby data are decoded from farther distances and a wider view angle with more capacity. COBRA\textsuperscript{23(24)} densifies data by aligning colored cells and utilizing color as another data channel. In addition, LightSync\textsuperscript{25} tackles the synchronization between the screen and camera, and RDCode\textsuperscript{26} enhances the reliability of communication. Furthermore, ShiftCode\textsuperscript{26} is a pattern-based shape-shifting barcode scheme with high throughput and reliability. These proposals offer high capacity and reliability, although they separate the content region for the user from the information region for the device. Consequently, meaningless code patterns impair the user’s visual experience, or render the information difficult to decode when the barcode is small. Nevertheless, these technologies are compatible with our study insofar as we propose a method for code imperceptibility whereas the researchers cited above propose specific code designs.

To resolve the code obtrusiveness issue, PiCode\textsuperscript{27} and ARTcode\textsuperscript{28} integrate images and code patterns. These proposals can be understood as utilizing the color channel for user-meaningful code rather than improved data capacity (as in, e.g., COBRA). Embedding data into an image in the spatial domain does not entail a specific working surface (e.g., paper or display), yet this
technique cannot in principle fully erase the visibility of the code. FoVLC\(^{29}\) improves data hiding capability adaptively based on the user's focal point, which however requires a display and an eye-tracking device, and the limitation of the visibility of the code remains in areas other than the focal point.

Embedding invisible code into images in the temporal domain overcomes this limitation. InFrame\(^{7}\) leverages high-speed luminance changes that are imperceptible to the human eye for encoding information acquired by cameras. This approach, however, requires a refresh rate of 120 Hz (where the CFF is about 60 Hz), and thus it is unclear whether this is an easily applicable technology in practical scenarios. Similarly, HiLight\(^{41}\) embeds bits into pixel translucency, modulating the \(\alpha\) channel of an image. Although HiLight works with displays that have a refresh rate of 60 Hz, the intensity of the \(\alpha\) changes should be very small in order to remain imperceptible. Other approaches include visual MIMO\(^{9}\), which embeds data using a Laplacian pyramid, and DisCo\(^{3}\), which transmits imperceptible signals with the display's backlight. Furthermore, some research projects reported improvements on InFrame or HiLight\(^{5,10,18}\). Our study differs these previous studies insofar as the latter emphasize the performance of the data transmission channel by using the full frame of the image and their original encoding. By contrast, we design the basic methodology applicable to other types of codes or marker patterns.

2.3 Screen–Camera Interaction

Some research seeks to facilitate the interaction between screens and cameras by using screens as an imperceptible data source. THAW\(^{20}\) tracks smartphones with a gradient-colored 2D pattern on the display that indicates the smartphone's position when captured by the camera. CapCam\(^{11}\) realizes display–camera pairing by detecting cameras with a capacitive touchscreen and transmitting data as a flashing color pattern. VR-Codes\(^{22}\) is a method of switching complementary colors and capturing them with a rolling shutter smartphone camera. Whereas this enables unobtrusive data transmission, it only produces output in grayscale and the code is exclusively embedded in the background of content. By maintaining the luminance of switching colors at a constant value, Yamamoto \textit{et al.}\(^{23}\) and our previous method\(^{19}\) embedded invisible data into an arbitrary image with a 60 Hz display. Yamamoto \textit{et al.} embedded random dot markers to track a camera device, mainly for use with an augmented reality system. In our previous method, each pixel corresponds to a data point decoded by a color sensor system. These methods are similar to ours, but differ in that they focus on the interaction, rather than the imperceptible marker embedding itself. Furthermore, these systems do not offer decoding with off-the-shelf devices like smartphones.

Although we have proposed the near-screen mobile interaction with imperceptible AR Markers by using our screen–camera system\(^{14}\), we have never explored the design scheme of embedding the 2D binary pattern by using the imperceptible color vibration.

2.4 Information Hiding

Several techniques are available for hiding information in digital media like images, audio, and video, such as steganography and watermarking. Steganography seeks to convey secret data in an ostensibly normal file, where the main concern is detectability\(^{39}\). Watermarking is technology that mostly embeds copyright information, prioritizing robustness to attacks (e.g., image compression or cropping), and sometimes sacrificing imperceptibility\(^{33}\). Most of these information-hiding techniques are achieved through the frequency domain using a discrete cosine transform or a Fourier transform, where decoding speeds are not a concern. Our study has different goals regarding transmitting and receiving data, viz., communicating in real-time, rather than hiding data or authenticating it. As a watermarking scheme, Kaleido\(^{36}\) is a piracy-preventing system that utilizes color switching. Kaleido is similar to our system in terms of its method, but with different goals (viz., generating noise in cameras rather than transmitting understandable data).

3. Encoding and Decoding Methods

Chromatic flicker, which is imperceptible to humans at frequencies above 25 Hz\(^{15}\), refers to high-speed color changes under constant luminance. Displaying two colors with the same luminance alternately at 60 Hz generates a chromatic flicker of 30 Hz. Thus, imperceptible color vibration can be generated at the operation frequency of common displays.

We modulate the original color of each pixel using two colors to represent a barcode pattern. A display transmits this imperceptible matrix barcode, and a smartphone camera acquires it for decoding through image processing, as illustrated in Figure 1. For instance, by switching between the two images in Figure 2, the QR code pattern shown in Figure 3 is extracted. Figure 4 shows the processing flow of the encoding and decod-
Fig. 2: Lena images with embedded barcode as color vibration. These QR code patterns are imperceptible when displayed alternately at 60 Hz or higher.

Fig. 3: Restored image from the embedded barcode with a smartphone. This will be converted to QR code.

Fig. 4: Processing flow of encoding and decoding methods

3.1 Encoding

(1) Pixel modulation

As stated above, we produce two modulated colors with the same luminance as the original color. In the XYZ color space, where Y value corresponds to luminance, we fix the Y value and search in the XZ plane for the modulated colors. They are located at symmetric positions with respect to the original color such that it is expected that they will be perceived as the original color when fused.

For the vibration of the modulated colors to have sufficient amplitude to be detected by the camera, we define a detectability function of the vibration between colors \((R_1, G_1, B_1)\) and \((R_2, G_2, B_2)\), as follows:

\[
E(R_1, G_1, B_1; R_2, G_2, B_2) = |EOTF(R_1) - EOTF(R_2)| + |EOTF(G_1) - EOTF(G_2)| + |EOTF(B_1) - EOTF(B_2)|
\]

\[
EOTF(x) = x^\gamma \ (x \in [0, 1])
\] (2)

Thus, each RGB value, in the range \([0, 255]\), is cast into \([0, 1]\), and cast back into \([0, 255]\) after the EOTF (electro-optical transfer function). We refer to the value of this function of the two modulated colors as the amplitude parameter \(r\) of the modulation.

In Equation (1) and (2), the EOTF denotes the gamma curve (\(\gamma\) is typically 2.2), which approximates the display’s nonlinearity of the input and output light intensity. The same input differences at different intensities are not outputted with the same differences, owing to the characteristics of the gamma, as illustrated in Figure 5a. Therefore, the amplitude in the physical world is correctly evaluated through the EOTF, as in (1) or as illustrated by Figure 5b, not by a naive numerical difference:

\[
|R_1 - R_2| + |G_1 - G_2| + |B_1 - B_2|
\]

(3)

By not considering this, there is non-uniformity to the vibration amplitude depending on the source intensity, resulting in decoding difficulties (Figure 6).

There remains a modulation parameter \(\theta\), denoting the direction of vibration and defined as the angle be-
between the line connecting the modulated colors and the x-axis. The optimal value of $\theta$ is unknown, and we further discuss this in Section 4.2 (1). From a brief experiment, we at least know that we should adopt a constant value in the whole image. Figure 7 shows a restored image where each pixel has a random $\theta$. The image is obviously unclear compared to that in Figure 3. This suggests that different $\theta$ among adjacent pixels cause color fusion and the attenuation of the vibration.

(2) Image modulation

The barcode pattern is represented by each pixel and assigned ‘0’ or ‘1’, the color for which we vibrate or not, respectively, using the modulation described in Section 3.1 (1). For instance, embedding a QR code into the Lena photo produces the two images shown in Figure 2.

Here, we need to narrow the gamut of the image to ensure a margin for vibration in the color space. Whether a color $(R, G, B)$ is usable can be determined by attempting modulation. However, it is not realistic to compute the usable gamut by trying all possible $(R, G, B)$ values. We approximate it by modulating the grayscale color $(v, v, v)$ ($v \in [0, 255]$) with various amplitudes $r$. For instance, the maximum and minimum $v$ when using $\theta = 0, \pi/2$ is computed as shown in Figure 8. Where $r = 80$ and $\theta = 0$, the maximum and minimum $v$ are 97 and 241, and we thus narrow the gamut of the image from $[0, 255]$ to $[100, 240]$, with some margin from these values. The lower range $(0, 96]$ in this case is not usable because the display gamma curve decreases the difference in low intensity input (Figure 5a).

The image gamut is narrowed simply by linear transform. With the maximum and minimum value of the narrowed gamut $k_{\text{max}}$ and $k_{\text{min}}$, a pixel value $k$ is replaced with

$$k_{\text{max}} - k_{\text{min}} \times \frac{k + k_{\text{min}}}{255}$$

This operation lowers the contrast of the image. We can reduce this effect by applying the contrast enhancement.
method beforehand, for example by equalizing the histogram, although this is optional because this method may produce visual artifacts such as contouring, mean-brightness change, and detail loss.

3.2 Decoding
Simply put, taking the differential between Figure 2 enables us to cancel the background and extract the code pattern. However, this cannot occur unless we consider the timing sequence of the display and camera, because alternating colors can be captured as a mixed color.

1. Camera parameters

We set the camera frame duration to $1/24 \text{s}$ (24 fps) and the exposure time to $1/120 \text{s}$. $1/120$ is half of the display’s refresh interval, and $1/24$ derives from $1/120 + 1/30$. This guarantees that pure colors and mixed colors appear alternately with an idle time of $1/30$ (vibration cycle) in between (Figure 9).

In addition, we need to unset any image-correction settings: viz., tone mapping and white balancing. Tone mapping is mainly for gamma correction, and typically applies the inverse function of $2$ such that the captured image will have linear intensity on the display. Since we set vibration amplitudes to be uniform by considering the display’s gamma, the camera tone mapping curve should be merely a line connecting $(0, 0)$ and $(1, 1)$. White balancing (WB) corrects image colors so that they are realistic under a different light source by adjusting a white object to be rendered white in the captured image. The Android platform has two parameters for this: gains and a transform matrix. The gains define the gains applied to Bayer raw color channels, and we set these to $[R, G_{even}, G_{odd}, B] = [2.0, 1.0, 1.0, 2.0]$, which results in simple demosaicing with the Bayer filter. The transform matrix is a color mixture matrix for RGB values, and we set this to the identity matrix. Although it is not mandatory to set WB parameters manually (i.e., automatic WB can be used), we fixed them to eliminate the device-dependent factor.

Finally, we set the focus distance and ISO sensitivity. These parameters depend on actual usage. There are two options for the focus distance: autofocus and manual focus. Autofocus is a useful function but sometimes causes continual focus changes and instability in decoding. By contrast, manual focus realizes stable decoding but requires either presetting the focus distance or user adjustment on the spot. Nevertheless, Experiments 4.2(5) and 4.2(6) showed good decoding results with autofocus enabled. The ISO sensitivity also needs to be set appropriately such that the sensors perceive the color vibration but do not saturate. The ISO depends on the color vibration amplitude $r$, and on the brightness of the display. These values are discussed in Experiment 4.2(4).

2. Image processing

Image processing to extract vibration patterns from the $i$-th captured frame ($F[i]$) is described below.

1. Take the absolute differential between the second previous frame.

$$D[i] = |F[i] - F[i-2]|$$

2. Add the previous differential to the current one.

$$A[i] = D[i] + D[i-1]$$

3. Noise reduction by blur and inverse-binarization.

$$O[i] = IBinarize(Blur(A[i]))$$

A simple illustration of this procedure is shown in Figure 10a. Each process is elaborated in the following paragraphs.

As seen in Figure 10a, Equation (5) aims to differentiate two pure colors. However, two mixed colors may also be differentiated. Letting the amplitude of pure colors be $1$ and the purity of the mixed colors be $p$ (completely pure: $p = 1$, completely mixed: $p = 0$), array $D$ will be $[2, 2p, 2p, \ldots]$ on vibrating pixels. The value $2$ should be sufficient, but $2p$ is not necessarily sufficient. Here, the absolute differential between the two frames $F1$ and $F2$ is defined as

$$|F1 - F2|(i, j) = |F1U(i, j) - F2U(i, j)|$$

$$+ |F1V(i, j) - F2V(i, j)|,$$

where $F_U, F_V$ denotes the U, V value of frame $F$ in YUV format, and $(i, j)$ denotes the pixel location. We use only the U and V value since we vibrate only chromaticity. It is true that Y (luminance) can also vibrate.
For each color

(a) For each color

(b) Considering the rolling shutter effect

Fig. 10: Image processing for decoding. Vibrating colors are processed as in (a), but actually like (b), considering intra-frame variance. The first row corresponds to array $F$, the second to $D$, and the third to $A$. $p$ denotes the purity of the mixed colors.

owing to a mismatch between display and camera characteristics, but we consider it to have more noise than the signal.

Equation (6) contributes to stabilizing the output. By adding the previous $D$, array $A$ will be \{2 + 2p, 2 + 2p, 2 + 2p, 2 + 2p, ...\}. This takes advantage of an insufficient output 2$p$, which could otherwise be wasted. Furthermore, this eliminates the temporal aliasing caused by the rolling shutter effect. Most smartphone cameras use CMOS sensors and implement line-scan frame acquisition and hence the exposure starts at a different time depending on the row in the sensor array. Owing to this rolling shutter, for instance, the upper part of the frame captures the pure color and the lower part of the frame captures the mixed color (Figure 11). If we do not use the stabilized $A[i]$, a single $D[i]$ has different values depending on the row, resulting in a vacancy of extracted code (Figure 12). The actual processing stream is illustrated in Figure 10b.

Now that we are able to acquire code patterns from the color vibration, Equation (7) simply reduces the noise and converts it to a black-and-white image. We adopt $7 \times 7$ blur filtering for noise reduction, because median filtering, which is desirable in this case for its edge-preserving feature, incurs considerable computational time. During inverse-binarization, the threshold is computed using Otsu’s method, and each pixel value is replaced with black if it is larger than the threshold and white if less.

We implemented the procedure above using RenderScript, a framework for running computationally intensive tasks at high performance on Android with parallelization. One frame must be processed within 41 ms ($\approx 1/24 \text{ sec}$) since we adopt 24 fps capturing.

4. Experiments

Our system has two requirements: that color vibration is not perceived to the human eye, and that the vibration is detectable by the camera. We thus conducted two experiments to confirm these respective requirements.

4.1 Color Perception

(1) Issue

Our key technique for imperceptibility is based on constant luminance. Yamamoto et al. revealed that, in the $L^*a^*b^*$ color space, vibrating only the $a^*$ or
b* value is less perceptible than vibrating only the L* value. The question remains whether to use the XYZ or L*a*b* color space. The main difference between them is that XYZ is linear to RGB and L*a*b* is non-linear (but perceptually uniform). This question thus reduces to the question of which color space reproduces the correct target colors when fused in temporal switching.

(2) Condition
To compare the target color, the fused color in the XYZ color space, and the fused color in the L*a*b* color space, we measured them with a spectrometer (UPRtek, MK350N Premium) put 50 mm in front of a color-calibrated liquid crystal display (CG-2420, EIZO); its luminance was 60 cd/m² in an assembled darkroom (ADR-F2, ASONE), into which no ambient light could enter. The target colors were Black (100, 100, 100), Gray (170, 170, 170), White (240, 240, 240), Red (240, 100, 100), Green (100, 240, 100), Blue (100, 100, 240), Cyan (100, 240, 240), Magenta (240, 100, 240) and Yellow (240, 240, 100). These (R, G, B) values were selected so that we were able to modulate them with r = 80 (see Section 3.1 (2) ). They were modulated by changing each axis value, X, Z, a*, or b*. A static single colored image was displayed to measure the target color, and two single colored images switching at 60 Hz were displayed to measure the fused color. All the images were 1920 × 1080. The integration time of the spectrometer was set to the largest multiple of 1/30 = 33.33 μs such that the spectrometer was not overexposed. Measurements were performed 10 times for each color and the average value was computed as the result.

(3) Results
The spectrometer outputted colors in X, Y, Z values. We computed color differences between a target color and fused colors based on the previous method\(^7\) that maps the values to the L*a*b* color space and taking the Euclidean distance. Table 1 shows the results, where we see no regularities. In fact, no significant difference was obtained from the Wilcoxon signed-rank test. Thus, we conclude that we can adopt the XYZ color space rather than the L*a*b* color space, insofar as the latter requires additional computational time.

### Table 1: Color difference of the target color and fused color modulated with each axis

|     | X    | a*  | Z    | b*  |
|-----|------|-----|------|-----|
| Black | 2.27 | 1.12 | 3.94 | 3.49 |
| Gray  | 0.261| 0.238| 0.348| 0.855|
| White | 0.403| 0.767| 0.558| 0.278|
| Red   | 0.252| 0.464| 2.90 | 2.27 |
| Green | 0.160| 0.375| 1.56 | 1.38 |
| Blue  | 1.31 | 0.810| 0.502| 0.644|
| Cyan  | 0.228| 0.312| 0.494| 0.333|
| Magenta| 0.00367| 0.581| 0.120| 0.333|
| Yellow| 0.294| 0.418| 1.57 | 1.22 |

Fig.13: QR code embedded in the experiments, containing text “https://www.nae-lab.org/” (192 bits), having error correction capability level L (the lowest), and composed of 25 × 25 cells.

4.2 Image Restoration
As a matter of course, we need to be able to restore data patterns from modulated images. We measured the decoding rate (the percentage of correctly decoded frames among all the captured frames) on images where we embedded a matrix barcode under some conditions, as shown in Figure 13. The decoding process was run for 30 seconds, which amounted to 720 captured frames. Here, when the decoding rate is p, the time required to receive the information is \((1/p) \times (1/24)\) s, in accordance with a 24 fps camera. For instance, a decoding rate of 40% requires only about 0.1 s. The experimental setup was the same as that described in Section 4.1 (2) except that the smartphone (Galaxy S8, Samsung) was placed perpendicular from the display instead of the spectrometer.

(1) Single color image
First, we targeted a 1920 × 1080 px single color image with a 540 × 540 px barcode as the simplest case. We chose θ as an independent variable since, as mentioned in Section 3.1 (1); its optimal value is unknown. There were nine target colors, as in Section 4.1 (2), and these were modulated with \(r = 80\). The smartphone was placed 30 cm away from the display. The focus distance was set to 30 cm, and ISO sensitivity was 500. We set these parameters as defaults. That is, we used them in the following experiment without a particular reference: e.g., image size, barcode size, amplitude \(r\), distance between smartphone and display,
focus distance, and ISO sensitivity.

The results are shown in Table 2. The results indicate that $\theta = \pi/2$ outperforms other $\theta$ values except when the target color is Gray or White, where $\theta = 0$ is the best. We compared images restored from Magenta with $\theta = \pi/4, \pi/2$, recording 17.4%, 92.1%, respectively (Figure 14). The captured amplitude with $\theta = \pi/4$ is obviously weaker than that with $\theta = \pi/2$, which showed a worse decoding rate. This suggests that vibration with $\theta = \pi/2$ is more recognizable by the camera than other directions.

However, the decoding rate was still low in the case of White, Green, and Yellow with all $\theta$. Figure 15a shows the restored image from Yellow with $\theta = \pi/2$. Compared to Figure 14b, we can observe the Moire pattern of the display, which functions as noise when decoding. Generally speaking, the Moire pattern occurs when two ruled patterns are overlaid on another similar pattern. In the case of an LCD and smartphone, an array of pixels interferes with an array of light sensors to generate unwanted artifacts (Figure 16a). Figure 16b depicts two arrays of lines showing a spatial-beat phenomenon.

Fortunately, the Moire noise is practically no problem, because it can be eliminated by tilting or panning the smartphone. However, to evaluate the effect of Moire noise elimination, we set the smartphone to have a 45° roll angle as a typical condition (Figure 16c). When the smartphone has this angle, it produces a clear image, as shown in Figure 15b. Our conclusion that $\theta = \pi/4$ and $\theta = 3\pi/4$ are less decodable owing to vibration weakness, and that brighter colors such as Yellow generate Moire noise, promoted us to conduct an additional experiment. We placed the smartphone

| $\theta$ = $\pi/4$ | $\theta$ = $\pi/2$ | $\theta$ = $3\pi/4$ |
|-------------------|-------------------|-------------------|
| Black             | 84.3              | 67.8              | 100               | 98.1              |
| Gray              | 67.8              | 13.1              | 58.5              | 36.7              |
| White             | 6.74              | 0.00              | 1.39              | 0.00              |
| Red               | 79.3              | 49.0              | 95.9              | 85.0              |
| Green             | 1.25              | 0.00              | 10.7              | 4.03              |
| Blue              | 74.4              | 68.1              | 98.6              | 84.4              |
| Cyan              | 23.2              | 0.28              | 46.5              | 16.1              |
| Magenta           | 75.4              | 17.4              | 92.1              | 39.4              |
| Yellow            | 0.139             | 0.00              | 0.417             | 0.139             |

Fig. 14: Restored images from Magenta. The decoding rates with $\theta = \pi/4, \pi/2$ were 17.4%, 92.1%, respectively.

Fig. 15: Restored images from Yellow with $\theta = \pi/2$. The decoding rates of the moire-noised pattern and the moire-eliminated pattern were 0.417% and 98.6%, respectively.

Fig. 16: Moire noise occurs because of interference between an array of pixels and an array of light sensors. (a) Difference between the alignments of display pixels and camera sensors. (b) A beating pattern is observed when two arrays of lines with different pitches are placed at the same angle. (c) Rotating the one array reduces this artifact, which, in this case, suggests that the smartphone should be placed at a 45° roll angle.
Table 3: Comparison of decoding rate of each $\theta$ for Moire-noised and Moire-eliminated

|       | Moire-noised |       | Moire-eliminated |
|-------|--------------|-------|------------------|
|       | $\theta = 0$ | $\theta = \pi/2$ | $\theta = 0$ | $\theta = \pi/2$ |
| Black | 84.3         | 100   | 98.6             | 100             |
| Gray  | 67.8         | 58.5  | 98.5             | 96.0            |
| White | 6.74         | 1.39  | 63.3             | 59.7            |
| Red   | 79.3         | 95.9  | 62.9             | 99.0            |
| Green | 1.25         | 10.7  | 74.0             | 91.8            |
| Blue  | 74.4         | 98.6  | 92.2             | 100             |
| Cyan  | 23.2         | 46.5  | 80.8             | 98.5            |
| Magenta | 75.4    | 92.1  | 81.3             | 100             |
| Yellow | 0.139       | 0.417 | 26.1             | 98.6            |

at a 45° roll angle and 35 cm away from the display with a 35 cm focus distance, where the target $\theta$ were 0, $\pi/2$. The decoding rates are shown in Table 3, where all the results are more than 50%, except Yellow with $\theta = 0$. We thus modified the default parameters to this setup (45° roll angle, 35 cm distance).

(2) General image

After confirming that data can be decoded from a single color image and that $\theta = 0$ or $\theta = \pi/2$ is promising, we conducted the same experiment on five general images (resized to 1920 $\times$ 1080) modulated with these two $\theta$. To test with a simple image, we created a mosaic image composed of single colored triangles (Figure 17a), whose colors are seven ones used in the SMPTE Color Bars. An abstract art image composed of solid colored squares, “Composition A” drawn by Piet Mondrian (Figure 17b), was also adopted for a simple case. Furthermore, as more complex images, we chose three standard test images (Lena, Mandrill, and Couple) from the USC-SIPI image database (Figure 17c-17e).

Table 4 shows the results. $\theta = \pi/2$ showed the better result in all cases, with the lowest decoding rate of 95.0%. This experiment proved that our system works well enough in practical applications, especially when using $\theta = \pi/2$. We thus include $\theta = \pi/2$ as a default parameter.

(3) Barcode Complexity

We here investigate how much impact the complexity of the barcode has on restored images. We targeted a Lena image embedded with various versions of a QR code. This version corresponds to the number of black-and-white cells in a barcode; namely, the higher the version is, the more capacity and complexity the barcode has. The information represented by the QR codes was the text composed of the repetition of the same text (https://www.nae-lab.org as in Figure 13 and all the error correction capacity was level L. The QR code in Figure 13 has the version 2, and Figure 18 shows the QR code with version 6, 10, and 12. Each of these QR codes has different information and code patterns.

The decoding rate according to the version of the QR code is in Figure 19. From the version 2 to the version 7, the decoding rate falls down as we naturally expect. Curiously, however, from the version 8 to the version 10, it fluctuates, which suggests that the factor affecting the decodability is not only the version (the smallness of cells). We consider that the design of the QR

![Fig. 17: General images in the experiment 4.2 (2)](a) Triangles (b) Artwork (c) Lena (d) Mandrill (e) Couple

![Fig. 18: QR codes with the version 6, 10 and 12 used in the experiment 4.2 (3)](a) Version 6 (b) Version 10 (c) Version 12

(n = 5, c = 41) (n = 11, c = 57) (n = 15, c = 65)
code, i.e., the alignment of cells, especially the density of the cells with the same color, has an effect. On the other hand, the decoding rates is higher than 30% for all versions below 10; thus, it seems that the threshold of the decodable version is 11 or 12 even considering these unstable values. We note that this threshold is possibly extended by the performance of the capture device.

(4) Vibration intensity

It is expected that the decoding rate depends first on the vibration amplitude \( r \). Second, the higher ISO sensitivity is, the more easily the vibration can be detected. In addition, a brighter backlight display will generate stronger vibration, even with the same \( r \). To reveal the relationship among these three parameters that relatively determine the vibration intensity, we targeted a Lena image modulated with various \( r \), where the camera’s ISO sensitivity was 100, 300, 500, 700, 1000, and 1300 with 60 cd/m\(^2\) display luminance, and with an ISO of 100, 300, 500, and 700 with 120 cd/m\(^2\).

The decoding rate with 60 cd/m\(^2\) is plotted in Figure 20a. Where ISO = 300, 500 and 700, the decoding rate naturally increased as \( r \) increased, with a steep slope at around \( r = 50 \). This means that \( r \) will suffice provided it is just higher than some threshold where decoding performs well (e.g., \( r = 60 \)). Where ISO = 100, the rate increased slowly and unstably. This is because an ISO sensitivity of 100 is too low to capture color vibration satisfactorily. Where ISO = 1000 or 1300, by contrast, is so high that some parts of the captured frames are saturated. Consequently, the rate reaches a ceiling before 100 % where ISO = 1000 and are bounded to 0 % throughout with all \( r \) where ISO = 1300.

The result with 120 cd/m\(^2\) is shown in Figure 20b. The decoding rate where ISO = 100 or 300 is more decodable than that in Figure 20a. However, there is little difference where ISO = 500 in 20a and ISO = 300 in 20b compared to where ISO = 300 in 20a and ISO = 100 in 20b, respectively. Where ISO = 500 and 700, saturation occurs, as with ISO = 1000 and 1300 at 60 cd/m\(^2\).

This experiment shows that we need to adjust the ISO sensitivity to a certain level depending on the screen brightness and, once it is adjusted, the performance hardly changes. This suggests that we should set \( r \) to around 60 to 90 and change the ISO sensitivity manually (or automatically, if possible) for practical use.

(5) Distance

Next, we determined how far from the display the smartphone can be. We targeted a modulated Lena image where the smartphone was placed at various distances with autofocus enabled. We tested four cases with varying barcode sizes: 135 × 135, 270 × 270, 540 × 540, and 1080 × 1080.

We plot the result in Figure 21. In the case of a 540 × 540 barcode, the decoding rate retained 40% at 73.64 cm. This shows that our system works at the distance five times farther than the marker size, given the fact that the barcode’s side length is 14.59 cm on this display. This means that when using an 80-inch display, our system can decode patterns about 5 meters away from the display because the marker can be
displayed at a size of about 1 m x 1 m. Compared to the case of a 270 × 270 barcode, the distance for the same rate is a little shorter than double the distance, which holds true between 540 × 540 and 1080 × 1080. This is accounted for by the effect of barcode scaling minus vibration attenuation. Nevertheless, this result shows that the larger the barcode is, the farther it can be decoded from.

However, the 135 × 135 case recorded an unexpected result, with a rate of 0 % when closest, and of 27.9 % at the highest, although the latter itself is not a bad value. We observed a noisy image when the display and smartphone were too close.

The barcode is captured within small pixels at a long distance, which may decrease of decoding rate. For example, the blurring filter of such a kernel size that is suitable at a middle range will be unsuitable at long distances. Thus, we conducted an additional experiment, where we targeted a Lena image with a 540 × 540 barcode using digital zoom for the captured barcode size to be fitted for decoding. We also plot this result in Figure 21, which indicates that processing the barcode at a more appropriate size gives us a better decoding rate.

(6) Applicability

Finally, we needed to demonstrate our system on several transmitters and receivers in a practical situation. We selected three displays—a 24.2-inch LCD monitor (CG2420, EIZO), a 15-inch laptop screen (Latitude E5570, Dell), and a 50-inch large display (50Z10X, TOSHIBA)—which we tested with three smartphones: a Galaxy S8 (Samsung), a OnePlus 5T (OnePlus), and an Xperia XZ1 (SONY). For all 3 × 3 combinations, we targeted a Lena image under indoor lighting with the smartphone held in one hand stably with autofocus enabled. We modified the blurring filter size to 7 × 7 to 3 × 3 with the Xperia XZ1 because the captured image was less noisy, though a bit blurred.

Table 5 lists the decoding rates. All the rates were acceptable, although the Xperia XZ1 on CG2420 was relatively low. Overall, the performance was better in the order of the Galaxy S8, OnePlus 5T, and Xperia XZ1, which suggests that there is device-dependency. From our observation, the rate depends on the performance of the camera sensor, such as how clearly or brightly it captures the image. Indeed, a decoding rate of 30.1 % for the Xperia XZ1 and CG2420 was acceptable, insofar as it still required only about 0.14 ms to decode on average. In conclusion, this experiment demonstrates the applicability of this system on multiple devices under practical situations.

5. Application

We here discuss three application scenarios for our system: advertising, tour guiding, and a movie trailer. Each of these scenarios is typical as a function of a public display and is a very good example of exploring an applicability for Human-Computer Interaction. In addition, we examined imperceptibility of embedded barcodes and usability of the applications through a user study using the advertisement scenario.

5.1 Application Scenarios

(1) Advertisement

The proposed approach can be employed for advertising, as shown in Figure 22. For instance, we can display product images while encoding the corresponding URLs for shopping. Passers-by see only the untainted images, and those interested in the product can access the website immediately by pointing their smartphone cameras at the displayed advertisement. This application enables advertisers to offer an appealing (and unobstructed) visual impression of their products that customers can purchase immediately.

(2) Tour Guiding

Figure 23 shows another possible application: tour guiding. For instance, in a train station, a map indicating landmarks can be embedded with additional related information available online. Users are able to see and know more about the landmark to decide whether it is worth visiting. Since the data is embedded in each area
and broadcasted, this allows multiple users to obtain information in accordance with their respective interests. Here, were the barcodes visible (e.g., with QR codes), placing several of them on a map would severely affect the user’s experience.

(3) Movie Trailer
Further, we can apply this approach to video by duplicating the frames and modulating them with the method presented here. We confirmed high-quality and data-decodable videos. This video-based approach can be used in applications such as embedding ticket reservation links within movie trailers (Figure 24).

5.2 User Study
To evaluate imperceptibility (RQ1), ease of use (RQ2) and usefulness (RQ3) of the proposed technique, we conducted a user study under the advertisement scenario, which is one of the applications above. The participants were ten university students (seven males and three females) with an average age of 24.0 (SD=3.0).

We reproduced a digital signage with an LCD (CG2420, EIZO). The monitor was installed at the eye level, on which four images of products (a jacket, a watch, a jean, and a shoe) were displayed as in Figure 22a. URLs for a web catalog corresponding to each product were embedded with the imperceptible QR code. The participant were instructed to hold the smartphone over the product images and browse the linked web catalogs as in Figure 22b. After that, they answered a questionnaire for verifying imperceptibility of the QR codes and SUS (System Usability Scale) test, which quantifies ease of use and usefulness of the system.

The concrete instructions were as follows: “We assume this LCD monitor is a digital signage installed at the passage in the station and that the imperceptible QR code reader application is installed in your smartphone. There is a white frame on the smartphone’s...”
screen, in which the imperceptible QR code will be detected by holding it over the product image. You can access a web catalog of each product using this reader. So let’s actually access the two catalogs of the jean and the shoe.”

To validate imperceptibility of the proposed technique, the following two questions are used “Did you directly see the QR codes embedded into the LCD monitor with your eye? (not through a smartphone)”, “Did you directly see display flicker with your eye? (not through a smartphone)”. As a result, all the participants answered “No” to both the questions; hence we argue that the proposed technique keeps imperceptibility (RQ1).

System usability was measured using the ten questions in the SUS. The participants’ answers in the 5-point Likert scales were transformed to a 0-100 scale to indicate the usability of the smartphone application. The mean SUS score was 71.25 (SD=11.50), which indicates that the participants perceived the usability of this application as “good” in the digital signage scene (RQ2, RQ3). We thus conclude that the proposed system provides ease of use and usefulness sufficient in ubiquitous computing environments.

6. Discussion

In this section, we address topics that remain for discussion.

We switch the chromaticity of each pixel while most existing methods switch the luminance, to earn imperceptibility with a display of lower refresh rate. Aside from imperceptibility, however, the comparison of the image restoration performance with a luminance-based approach can be a concern. We believe that this difference in modulation does not have a significant impact on the performance; because we can use exactly the same protocol when we vibrate luminance, and the only possible factor affecting the performance is the camera’s sensitivity. Smartphone camera may be a bit more sensitive to luminance than chromaticity and thus may achieve better performance, but we consider that it is not a dominant factor in the performance.

Section 3.2 (1) discussed 24 fps and 1/120 s exposure as camera settings. We did not use 120 fps owing to restrictions with the Android platform, although this appears simple and effective. Currently, capturing images at an fps above 30 requires the use of a high-speed capture API, which causes problems. First, high-speed-captured images cannot be directly poured into the image-processing stream, and a video must be recorded and post-processed for the video file. This prevents decoding in real-time. Other problems include low image resolution and the inability to set the ISO sensitivity manually.

Regarding ISO sensitivity, in Experiment 4.2(4), adjustments were necessary, depending on the transmitter. Although automatic adjustment is obviously desirable, the built-in function is disabled once the frame duration (fps) or exposure time is manually modified on the Android platform. Currently, therefore, we need to develop a software function to adjust the sensitivity by feeding back the brightness of the captured image.

We address the applicability of this system, despite the good results in Experiment 4.2(6). We confirmed that some phones do not accept manual modifications of the camera parameters, as described in Section 3.2(1), and that our system does not work on these phones. These phones do not support full imaging capabilities, which we believe is due to cost reductions or poor hardware, although this should be resolved in the future. We implemented this system exclusively on the Android platform, but we expect that it can be used on other operating systems as well, such as iOS. If the camera can be configured to 24 fps and 1/120 s exposure and if it is capable of real-time image processing, this system should fully work in principle. If not, a workaround using 120 fps recording and post-image processing is feasible, although this would sacrifice real-time decoding.

We proved that our system works in practical situations, but only when the color vibration parameters are fixed at $r = 80$ and $\theta = \pi/2$. It is possible to improve the contrast of modulated images or the decoding rate by adaptively determining these parameters in consideration of the pixel intensity and the imperceptibility of code patterns for each image. At this time, the relationship between $\theta$ and decoding rate in each color (described in Section 4.2(1)) will provide a clue to determine the optimum values. If $r$ is decreased, the contrast of images is improved, but the decoding rate of the code pattern is lowered; thus, it is necessary to search for the minimum $r$ indicating a practical decoding rate. We set $r$ and $\theta$ to the same value throughout the code patterns, but it may be possible to improve the decoding rate by making these values different for each pixel. Since $r$ is defined in consideration of gamma correction, the amplitude of actual color vibration differs according to the luminance value of each pixel even
if the \( r \) is set to the same value. As we described in Section 3.1(1), setting \( \theta \) to a different value for each pixel in the embedded code patterns attenuates color vibrations received by the camera; therefore, we need to keep in mind that \( \theta \) must be set to the same value throughout the code patterns. In addition, although we discussed the relationship between the camera-display distance and the performance of our system in Section 4.2(5), we can further explore this relationship when we change the detail of code patterns.

Finally, we discuss bidirectional communication in our system. Our method focuses on transferring information from the screen to the camera device. In order to achieve bidirectionality, it is necessary to transmit information reversely from the device to the screen using some method. The combination of LTE or Wi-Fi and the internet is the most typical approach, and NFC and visible light communication with phone’s flashlight\(^{60}\) are also viable options since there is only a short distance between the display and the devices. All of these approaches can be implemented with only built-in peripherals in smartphones or tablets: they thus match well with our method. Therefore, we conclude it is relatively easy to realize bidirectional screen–camera communication.

7. Conclusion

We proposed embedding imperceptible matrix barcodes into colorful images by leveraging chromatic flicker and extracting it in real-time. Image restoration merely requires ordinary LCDs and off-the-shelf smartphones. We elaborated modulation parameters such as the gamma-aware amplitude \( r \) and static vibration direction \( \theta \), and we detailed the decoding process, including the display–camera timing sequence and rolling shutter avoidance. Our experiments showed that the XYZ color space is preferable over the \( \text{L}^*\text{a}^*\text{b}^* \) color space, and that \( \theta = \pi/2 \) is apt for this system. It also turned out that the design of barcode as well as the complexity affects image restoration. Moreover, the ISO sensitivity needs to be adjusted to within an appropriate range and the decoding distance is scalable.

Finally, we showed that our system works on multiple devices. To show how we envision the use of the system in the field of Human-Computer Interaction and ubiquitous computing, we presented concrete applications that match the usage in these research fields and examined imperceptibility of embedded barcodes and usability of the applications through a user study. Thus far, our study demonstrates the fundamental technique for imperceptible pattern embedding; that is, temporally dynamic data and robust barcode design, or another embedded pattern to realize another function are envisioned to enhance this system in future work.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers JP16H01739, JP17J04216, JP19J00101 and JST ACT-X Grant Number JPMJAX1900.

References

1) C. J. Yang and A. P. Sample, “EM-Comm: Touch-based Communication via Modulated Electromagnetic Emissions,” Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol.1, no.3, pp.1–24, sep 2017.
2) ITU-T/ISO/IEC, “Automatic identification and data capture techniques – QR Code 2005 bar code symbology specification,” 2006. https://www.iso.org/standard/62021.html
3) K. Jo, M. Gupta, and S. K. Nayar, “DisCo: Display-Camera Communication Using Rolling Shutter Sensors,” ACM Transactions on Graphics, vol.35, no.5, pp.1–13, jul 2016.
4) T. Li, C. An, X. Xiao, A. T. Campbell, and X. Zhou, “Real-Time Screen-Camera Communication Behind Any Scene,” Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys ’15, pp.197–211, 2015.
5) V. Nguyen, Y. Tang, A. Ashok, M. Gruteser, K. Dana, W. Hu, E. Wengrowski, and N. Mandayam, “High-rate flicker-free screen-camera communication with spatially adaptive embedding,” Proceedings of the 35th Annual IEEE International Conference on Computer Communications - INFOCOM ’16, pp.1–9, apr 2016.
6) S. Shi, L. Chen, W. Hu, and M. Gruteser, “Reading between lines: high-rate, non-intrusive visual codes within regular videos via ImplicitCode,” Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - Ubicomp ’15, pp.157–168, 2015.
7) A. Wang, C. Peng, O. Zhang, G. Shen, and B. Zeng, “InFrame: Multiflexing Full-Frame Visible Communication Channel for Humans and Devices,” Proceedings of the 13th ACM Workshop on Hot Topics in Networks - HotNets-XIII, pp.1–7, 2014.
8) A. Wang, Z. Li, C. Peng, G. Shen, G. Fang, and B. Zeng, “InFrame++: Achieve Simultaneous Screen-Human Viewing and Hidden Screen-Camera Communication,” Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys ’15, pp.181–195, 2015.
9) W. Yuan, K. Dana, A. Ashok, M. Gruteser, and N. Mandayam, “Dynamic and invisible messaging for visual MIMO,” Proceedings of the 2012 IEEE Workshop on the Applications of Computer Vision - WACV ’12, pp.345–352, jan 2012.
10) H. N. Ricciuti and H. Misiak, “The Application of the Constant Method in Determining Critical Flicker Frequency (CFF),” The Journal of General Psychology, vol.51, no.2, pp.213–219, oct 1954.
11) Y. Jiang, K. Zhou, and S. He, “Human visual cortex responds to invisible chromatic flicker,” Nature Neuroscience, vol.10, no.5, pp.657–662, may 2007.
12) S. Abe, A. Arami, T. Hiraki, S. Fukushima, and T. Naemura, “Imperceptible Color Vibration for Embedding Pixel-by-Pixel Data into LCD Images,” Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA ’17, pp.1464–1470, 2017.
13) S. Abe, T. Hiraki, S. Fukushima, and T. Naemura, “Screen-Camera Communication via Matrix Barcode Utilizing Imperceptible Color Vibration,” Adjunct Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology - UIST ’18 Adjunct, pp.166–168, 2018.
14) A. Matsumoto, S. Abe, T. Hiraki, S. Fukushima, and T. Naemura, “Imperceptible ar markers for near-screen mobile interaction,” IEEE Access, vol.7, pp.79927–79933, jun 2019.
15) P. H. Pathak, X. Feng, P. Hu, and P. Mohapatra, “Visible Light Communication, Networking, and Sensing: A Survey, Potential
and Challenges,” IEEE Communications Surveys & Tutorials, vol.17, no.4, pp.2047–2077, 2015.
16) Y. Tanaka, S. Haruyama, and M. Nakagawa, “Wireless optical transmissions with white colored LED for wireless home link,” Proceedings of the 11th IEEE International Symposium on Personal Indoor and Mobile Radio Communications, vol.2, pp.1325–1329, 2000.
17) C. Danakis, M. Afgani, G. Povey, I. Underwood, and H. Haas, “Using a CMOS camera sensor for visible light communication,” Proceedings of the 2012 IEEE Globecom Workshops, pp.1244–1248, dec 2012.
18) D. Schmidt, D. Molyneaux, and X. Cao, “PICOntrol: using a handheld projector for direct control of physical devices through visible light,” Proceedings of the 25th Annual ACM Symposium on User Interface and Software Technology - UIST ’12, pp.379–388, 2012.
19) T. Hiraki, S. Fukushima, Y. Kawahara, and T. Naemura, “Physical field: an integrated field with physical robots and digital images using projection-based localization and control method,” SICE Journal of Control, Measurement, and System Integration, vol.11, no.4, pp.302–311, 2018.
20) T. Hiraki, S. Fukushima, H. Watase, and T. Naemura, “Dynamic pvc: Pixel-level visible light communication projector with interactive update of images and data,” ITE Transactions on Media Technology and Applications, vol.7, no.4, pp.160–168, 2019.
21) S.D. Perli, N. Ahmed, and D. Katabi, “PixNet: LCD-camera pairs as communication links,” Proceedings of the 2010 ACM Annual SIGCOMM Conference - SIGCOMM ’10, vol.40, p.451, 2010.
22) T. Hao, R. Zhou, and G. Xing, “COBRA: color barcode streaming for smartphone systems,” Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services - MobiSys ’12, p.85, 2012.
23) Q. Wang, M. Zhou, K. Ren, T. Lei, J. Li, and Z. Wang, “Rain Bar: Robust Application-Driven Visual Communication Using Color Barcodes,” Proceedings of the IEEE 35th International Conference on Distributed Computing Systems, pp.537–546, jun 2015.
24) W. Hu, H. Gu, and Q. Pu, “LightSync: unsynchronized visual communication over screen-camera links,” Proceedings of the 19th Annual International Conference on Mobile Computing and Networking - MobiCom ’13, p.15, 2013.
25) A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, and G. Shen, “Enhancing reliability to boost the throughput over screen-camera links,” Proceedings of the 20th Annual International Conference on Mobile Computing and Networking - MobiCom ’14, pp.41–52, 2014.
26) T. Zhan, W. Li, X. Chen, and S. Lu, “Capturing the Shifting Shapes: Enabling Efficient Screen-Camera Communication with a Pattern-based Dynamic Barcode,” Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol.1, no.1, pp.1–25, mar 2018.
27) W. Huang and W.H. Mow, “PicCode: 2D barcode with embedded picture and ViCode: 3D barcode with embedded video (poster),” Proceedings of the 19th Annual International Conference on Mobile Computing and Networking - MobiCom ’13, p.139, 2013.
28) Z. Yang, Y. Bao, C. Luo, X. Zhao, S. Zhu, C. Peng, Y. Liu, and X. Wang, “ARCode: preserve art and code in any image,” Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp ’16, pp.904–915, 2016.
29) O. Yildiz and B. Gulbahar, “Fovlc: Foveation based data hiding in display transmitters for visible light communications,” Proceedings of the 14th International Wireless Communications Mobile Computing Conference - IWCMC ’18, pp.629–635, 2018.
30) S.W. Leigh, P. Schoessler, F. Heibeck, P. Maes, and H. Ishii, “THAW: tangible interaction with see-through augmentation for smartphones on computer screens,” Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI ’14, pp.89–96, 2015.
31) R. Xiao, S. Hudson, and C. Harrison, “CapCam: Enabling Rapid, Augmented, Position-Tracker Implantation Between Devices,” Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces - ISS ’16, pp.169–178, 2016.
32) G. Woo, A. Lippman, and R. Raskar, “VRCodes: Unobtrusive and active visual codes for interaction by exploiting rolling shutter,” Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality - ISMAR ’12, pp.59–64, nov 2012.
33) G. Yamamoto, L. Sampaio, T. Taketomi, C. Sandor, H. Kato, and T. Kuroda, “Imperceptible On-Screen Markers for Mobile Interaction on Public Large Displays,” IEICE Transactions on Information and Systems, vol.E100.D, no.9, pp.2027–2036, 2017.
34) A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, “Digital image steganography: Survey and analysis of current methods,” Signal Processing, vol.90, no.3, pp.727–752, mar 2010.
35) V.M. Potdar, S. Han, and E. Chang, “A survey of digital image watermarking techniques,” Proceedings of the 12th IEEE International Conference on Industrial Informatics - INDIN ’05, pp.709–716, 2005.
36) L. Zhang, C. Bo, J. Hou, X.-Y. Li, Y. Wang, K. Liu, and Y. Liu, “Kaleido: You Can Watch It But Cannot Record It,” Proceedings of the 21st Annual International Conference on Mobile Computing and Networking - MobiCom ’15, pp.372–385, 2015.
37) K. McLaren, “The Development of the CIE 1976 (L*a*b*) Uniform Colour Space and Colour-difference Formula,” Journal of the Society of Dyers and Colourists, vol.92, no.9, pp.338–341, 1976.
38) SMPTE, “Eq 1:990 - smpte engineering guideline - alignment color bar test signal for television picture monitors,” EG 1:990, pp.1–3, March 1990.
39) J. Brooke, “SUS - a quick and dirty usability scale,” Usability evaluation in industry, vol.189, no.194, pp.4–7, 1996.
40) M. Otkan, C. Kray, and G. Povey, “Flexible joint optical interaction between smartphones and public displays,” Proceedings of the 4th International Symposium on Pervasive Displays - PerDis ’15, pp.39–46, 2015.

Satoshi Abe received B.S. and M.S. degrees from the University of Tokyo, Japan, in 2017 and 2019, respectively. His research interests include augmented reality, human–computer interaction, and ubiquitous computing. He can be contacted at abe@nae-lab.org.

Takefumi Hiraki received a Ph.D. in Information and Communication Engineering from the University of Tokyo, Japan, in 2019. He is currently an Assistant Professor at the Interfaculty Initiative in Information Studies, the University of Tokyo. His research interests include augmented reality, human–computer interaction, and ubiquitous computing. He is a member of IEEE and ACM. He can be contacted at takefumi@nae-lab.org.

Shogo Fukushima received the Ph.D. in engineering from the University of Electro-Communications, Japan, in 2013. He is currently an Assistant Professor with the Interfaculty Initiative in Information Studies, the University of Tokyo. He was a Visiting Assistant Professor at the Graduate School of Engineering Science, Osaka University. His research interests include virtual reality, entertainment computing, and human emotions. He is a member of the ACM. He can be contacted at shogo@nae-lab.org.

Takeshi Naemura received a Ph.D. in Electrical Engineering from the University of Tokyo, Japan, in 1997. Currently, he is a Professor in the Interfaculty Initiative in Information Studies, the University of Tokyo. He is a visiting assistant professor of computer science at Stanford University, USA, supported by ‘Japan Society for Promotion of Science (JSPS) Postdoctoral Fellowships for Research Abroad’ from 2000 to 2002. His research interests include virtual reality and human interface. He is a member of IEICE, IEEE, and ACM. He can be contacted at naemura@nae-lab.org.