# InFrame: Multiflexing Full-Frame Visible Communication Channel for Humans and Devices

Anran Wang[1], Chunyi Peng[2], Ouyang Zhang[2], Guobin Shen[3], Bing Zeng[4]
[1]Beihang University, [2] The Ohio State University, [3]Microsoft Research, China;
[4]University of Electronic Science and Technology of China
wangar@act.buaa.edu.cn; {peng.377, zhang.4746}@osu.edu; jackysh@microsoft.com; eezeng@uestc.edu.cn

## ABSTRACT

Recent efforts in visible light communication over screen-camera links have exploited the display for data communications. Such practices, albeit convenient, have led to contention between space allocated for users and content reserved for devices, in addition to their aesthetic issues and distractive nature. In this paper, we propose INFRAME– a system that enables *dual-mode* full-frame communication for both humans and devices *simultaneously*. INFRAME leverages the temporal flick-fusion property of human vision system and the fast frame rate of modern display. It multiplexes data onto full-frame video contents through a novel complementary frame design and several other techniques. It thus ensures screen-camera data communication without affecting the primary video-viewing experience for human users. Our preliminary experiments have confirmed that INFRAME can achieve about 12.8kbps data rate with imperceptible video artifacts when being played back at 120FPS.

## Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: GeneralData communications; C.4 [**Performance of Systems**]: *Design Studies*

## General Terms

Design, Experimentation, Human Factors

## Keywords

Screen-camera link; Full-frame video; Dual-mode visible communication; InFrame

## 1. INTRODUCTION

The primary purpose of a display is to convey human perceptible information to human eyes. Recent years have witnessed the universal adoption of visible screen-to-camera links as a valuable side channel for device-to-device communication [1–4]. Dynamic barcodes, e.g., Quick Response (QR) codes, are thus introduced to increase the data-carrying capability over the visible channel between the screen and the video camera. However, due to the limited display space, human-friendly visual content (usually in the form of images and clips) and device-favorable visual content (e.g., QR codes) have contended for display space allocation and led to aesthetical challenges. We have consequently seen that a QR code can only take a small area of the entire space. This not only limits its information-carrying ability, but also induces extra efforts to capture codes (e.g., being close enough to QR codes). The user experience for display rendering is also deemed distractive. Therefore, a key question arises: Can we eliminate this contention by restoring full-frame viewing for users while simultaneously establishing full-frame screen-device data communication? Fundamentally, it calls for a novel paradigm of dual-mode communication, which enables concurrent delivery of primary video content and other information without impairing user-viewing experience.

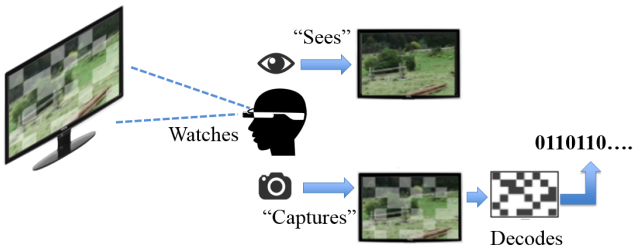



**Figure 1: Concept of full-frame multiplexing: screen-to-eye for videos and screen-to-camera for data communication over the same visible channel simultaneously.**

In this paper, we present INFRAME, a new system that enables dual-mode, full-frame, visible communication by multiplexing the visible channel to carry data over normal video content. Figure 1 illustrates the concept of full-frame multiplexing, where the screen shows the video content multiplexed from the original video and data frames (e.g., QR codes). The user can still watch the video as usual, without noticing the embedded data frames. The camera captures the data carried by the multiplexed frames and decodes the relevant information.

INFRAME is built upon the capability gap between human


*HotNets-XIII,* October 27–28, 2014, Los Angeles, CA, USA.

http://dx.doi.org/10.1145/2670518.2673862.

eyes and electronic devices. It is known that human vision system has its physical limit, whereas modern screens and cameras have well exceeded our eyes in terms of temporal resolution (*i.e.*, speed). Specifically, human vision system has up to 40-50Hz temporal resolution beyond which we cannot capture faster-moving objects. In contrast, modern displays, especially those 3D capable ones, support 120FPS or higher refresh rate. Smartphone cameras can take high resolution images at high frame rates. For example, Samsung Galaxy S5 offers 16-mega pixel resolution and 120FPS capture rate, while iPhone 6 supports 8-mega pixel resolution and 240FPS capture rate. Moreover, device capability is advancing at a much faster pace than human vision system can ever keep up with. We believe that the gap will grow further larger over time.

There are two challenges when designing full-frame dual-mode visible communication of INFRAME. First, the requirement to not affect the primary screen-eye channel imposes a rigid constraint to dual-mode communication. Human vision system is sensitive to flickers (see §2). We thus need to find an appropriate frame multiplexing scheme to combine a data stream and an arbitrary video clip. The goal is to let the presented video be perceived without color distortion, artifacts and flickers when being rendered on a high-frequency display. Second, the secondary screen-camera channel could experience potential interference from the original video content on the primary channel. We need to design an effective data frame encoding scheme. It should be resilient to both video content changes and known screen-camera communication limitations (e.g., frame rate mismatch, rolling shutter effect, poor capture quality).

In INFRAME design, we overcome the first challenge by leveraging the flicker-fusion property of human vision system, *i.e.* the exhibition of a low-pass filter effect when viewing scenes that change faster than critical flicker frequency, and the superior temporal resolution of displays. We come up with a key *complementary frame* concept and embed a data frame into a pair of multiplexed video frames (§3.2). A multiplexed frame will have obvious artifacts, but when played fast enough, the artifacts on complementary frames will cancel each out. We further propose *block-smoothing* technique to gradually embed data frames into video contents. To combat the second challenge, we design a hierarchical data frame structure and also special encoding and decoding schemes (§3.3). In particular, the encoding scheme uses the on/off states of an artificial chessboard pattern to signal a 0/1 bit. The decoding scheme detects the existence of chessboard pattern and recovers carried bit by checking induced noise levels. The encoding and decoding schemes together makes INFRAME resilient to video content.

We have implemented the INFRAME system and conducted a preliminary study. Our experiments confirm that INFRAME enables dual-mode full-frame visible communication to both humans and devices simultaneously. Without noticeable artifacts or flickers, it offers up to 12.8 kbps for a Lumia 1020 phone using a pure light gray video displayed on a 24' LCD monitor, and about 7.0 kbps when being multiplexed over a normal video. To the best of our knowledge, INFRAME is the first system to support *simultaneous, full-frame* video viewing and data streaming between a screen, a user and a device. Though the achievable throughput does not sound that appealing at the moment (but still comparable to that in other proposals [1–4]), it holds promise for further improvement with more research effort and increasing device capabilities.

## 2. BACKGROUND ON HUMAN VISION

Visual perception is critical to people. We obtain about 80% of information from the physical world through our eyes. Human vision system consists of foveal (also called central) and peripheral vision. Fovea is in the middle of the inner retinal surface. It is responsible for sharp central vision, essential to capturing important visual details such as reading and driving. Peripheral vision is part of the vision that occurs outside the center of gaze. It is good at detecting motion, and becomes more effective in the dark.

The structure of our vision system is similar to that of a camera [5]. Compared with cameras, our eyes also have lens (crystalline lens) and sensor (retina), but do not contain a shutter. As a result, there is no "exposure" process when we see things. Moreover, our visual perception possesses two features of *low-pass flicker fusion* and *phantom array effect*.

**Low-Pass Flicker Fusion** Human eyes have a special characteristic of *flicker fusion*, where beyond a certain frequency, termed *critical flicker frequency* (CFF for short hereafter) [6], time-variant fluctuations of light intensity are not perceptible to human eyes [7]; Instead, human eyes only perceive the average luminance. Vision research results in 1950s-1970s [7–11], have reported that CFF of human eyes is about 40-50Hz in typical scenarios. Consequently, flickers presented by a 60Hz CRT monitor are not perceptible. The temporal behavior of human vision system can be approximated as a *linear low-pass filter* at a high frequency exceeding the CFF, while CFF is affected by many factors including color contrasts, motion, luminance waveforms, *etc.*.

**Phantom Array Effect** It describes another special property of human eyes – sensitive to motion. While fast-moving objects zoom across view (either by object motion, or by eye motion such as rolling eyes), flicker can be noticed by the observer even when the display frequency is much higher [12]. For example, a fast flashing LED moving in a dark environment can be observed. It implies that CFF can be higher due to the *phantom array effect*. Unlike flicker fusion, the origin of phantom array effect is not fully understood. Recent studies show that lower flicker amplitude, larger duty cycle and larger beam size make it less visible [13, 14].

## 3. INFRAME DESIGN

We aim to build INFRAME, which enables full-frame, concurrent communication for users and devices. A secondary

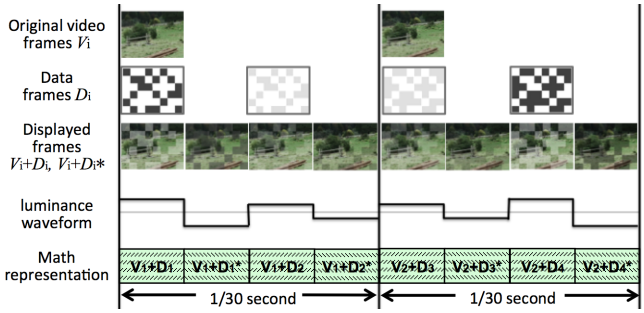



**Figure 2: Illustration of our overall design in INFRAME. We use a 120Hz display and deliver a 30FPS video. Four displayed frames are generated from one video frame and two data frames. $V_i$ and $D_i$ represent the $i$-th original video frame and data frame.**

data channel is established on top of the primary video content channel, without incurring interference to the video-viewing experience of the user, while still being able to transfer information. In our usage scenario, a camera captures data from the screen, while the user is watching a TV program without noticeable quality degradation.

The key idea of INFRAME is to exploit the perception gap between human vision and modern camera systems. Figure 2 illustrates the overall design. It ensures two design goals on screen-eye and screen-camera communications. First, INFRAME safeguards the normal full-frame video watching experience for users, while embedding data bits. Second, it enables data communication under the constraints imposed by the primary video channel.

## 3.1 Naive Design

Before we elaborate on each design component, we first describe some naive designs. Assume a 120Hz display and a 30FPS video for playback. Figure 3 illustrates several naive designs we initially came up with, all of which failed to address the first challenge with noticeable flickers. Figure 3(a) shows the original 30FPS video, and (b) illustrates its normal display when all video frames are used at the 120Hz refresh rate. Figure 3(c) illustrates a naive and aggressive scheme where three distinctive data frames are inserted after each video frame. We conducted a user study on its visual effect and found that this scheme incurred severe flickers. Dynamic semi-transparent data blocks are seen by human eyes. This is because it fails to meet the CFF requirement (*i.e.*, 40-50 Hz) due to the abrupt transitions between video frames and data frames. We modified how to insert video and data frames (*e.g.*, a video frame is followed evenly by a data frame in Figure 3(d)). We further tested other options on the V:D ratio (here, 2:2 and 3:1), data frame patterns and colors, and luminance levels. Unfortunately, their visual effect showed little improvement. For all such naive designs, obvious artifacts and color distortions were observed, because the average of sequential data frames did not match that of original video frames. The failure of these naive designs suggests that display frequency is not the only factor to cause abrupt change of video and data frames.

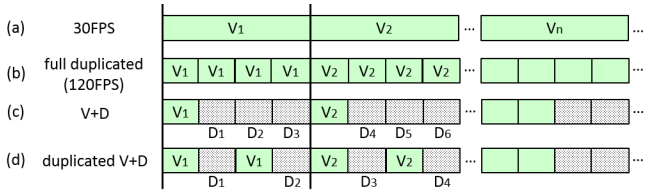



**Figure 3: Illustration of naive designs.**

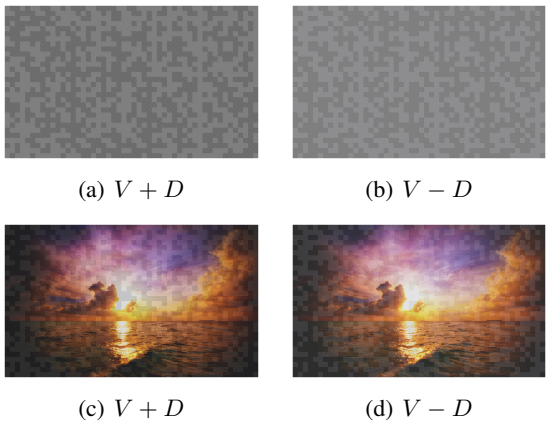
## 3.2 Screen-Eye Communication Design

To assure clean Screen-Eye communication (*i.e.*, a normal video-viewing experience), we propose *complementary frames* to leverage the low-pass filter effect on flicker fusion of the human vision system.

• **Complementary Frame Concept** By following the low-pass filter property of the human vision system, if we place two back-to-back video frames with one being the inverse of the other, we will perceive an average uniform color frame. This observation spawns the *complementary frames*, the core concept of INFRAME. We first define complementary pixels. Two pixels $p$ and $p^*$ complement each other with respect to the luminance level $v$ if their pixel values sum up to $2v$, *i.e.*, $v_p + v_{p^*} = 2v$. Consequently, two frames $P$ and $P^*$ are complementary frames with respect to the luminance level $v$ if all their pixels are complementary w.r.t $v$. It is easy to see that, after low-pass filtering, two complementary frames yield average frames with luminance level $v$. Complementary frames thus enable proper grouping of video frames and data frames (*e.g.*, V $\pm$ D) without incurring color distortion and noticeable change in the video's luminance level.

• **Leverage Flicker Fusion** We now present how to embed data into video frames. The lessons learnt from naive designs suggest that our human vision system be prone to flickers and visible intervention. Applying the complementary frame concept, we generate two data-plus-video frames ($V_i$+$D_i$, $V_i$+$D_i^*$) with respect to the original video frame $V_i$. As shown in Figure 2, given a 30FPS video stream $V$ and a data stream $D$, we duplicate each video frame $V_i$ four times (the refresh rate is 120Hz). Meanwhile, we fetch each data frame $D_i$, calculate its complementary frame $D_i^*$, and then embed both into the original video frame (i.e., $V_i + D_i$ and $V_i + D_i^*$, equivalent to $V_i \pm D_i$). Our study shows that, this scheme is effective on imposing negligible effect on video-viewing experiences. The maximum frequency of the waveform is 60Hz on a 120Hz display, which exceeds the CFF of human eyes and is thus imperceptible to human eyes.

• **Data Block Smoothing** A sequence of varying data frames $\{D_i\}$ might still impose vision interference when it sharply switches from one to another (e.g, from $V_1 + D_1^*$ to $V_1 + D_2$). This is because the abrupt switching (phantom array effects) causes flickers. To mitigate this effect, we devise a *block-smoothing* technique. We use one iteration to denote two complementary video+data frames (see Figure 5 for an illustrative example). We first increase the data cycle for each frame to ensure proper spatial block density, thus



(a) $V + D$ (b) $V - D$

(c) $V + D$ (d) $V - D$

**Figure 4: Examples of complementary frame pairs, where (a)(b) are using a pure color (gray) frame, and (c)(d) are using a normal video frame.**

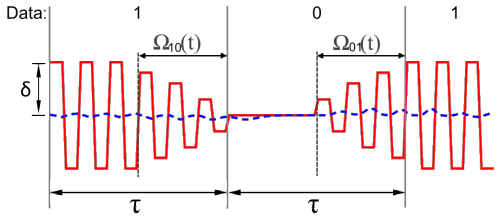



**Figure 5: Temporal smoothing waveform (red solid curve) adopted in InFrame and its effect (blue dotted curve) after applying an electronic low-pass filter.**

effectively increasing the beam size and duty cycle. Consequently, to deliver one data frame $D_i$, it may take $\tau$ ($>1$) iterations. Second, we gradually change the amplitude of the waveform of the temporal sequence if it needs to switch from 0 to 1 or vice versa. The amplitude of data frames is tuned by the parameter $\delta$, which determines the impact on the luminance level. The larger the value $\delta$, the bigger its effect on original video viewing. If the pixel is invariant (still 0 or 1) over consecutive data frames, the amplitude remains constant. When it switches from 1 to 0 or vice versa at the $\tau/2$-th iteration, the amplitude envelope follows a function $\Omega_{10}(t)$ or $\Omega_{01}(t)$ within the remaining $\tau/2$ iterations. This intends to smoothen the transition in each pixel in subsequent frames. In our experiments, we use half of the square-root raised Cosine waveform, after comparing with linear and stair function forms. Parameters $\tau$ and $\delta$ (to be elaborated in §3.3) are also critical to data communications.

In essence, we generate multiple intermediate data frames, which are multiplexed with video frames, to ensure imperceptible transition between two different data frames. We verified the design by passing the waveform to an electronic low-pass filter and observed stable output waveform. Note that, the above temporal smoothing also facilitates to mitigate the mismatch between the display refresh rate and camera capture rate. If data are changing too fast, the camera cannot capture all data, due to the rolling shutter effect [15]. With the smooth transition, we reduce interferences between adjacent data frames.

### 3.3 Screen-Camera Communication Design

A key difference exhibits between the screen-camera communication of InFrame and those conventional proposals that exclusively occupy one screen [1–4]. InFrame needs to realize bit representation under the constraints of the primary video channel (i.e., data is embedded over video frames). We now elaborate on the new design of $D_i$, as well as those features similar to barcodes for screen-camera communication. Assume the frame size as $w * h$.

• **Data Frame Structure** In our design, we merge $p * p$ pixels in the data frame $D_i$ to form one *super Pixel* (Pixel for short afterwards), in which Element pixels are assigned the same value. A Pixel is the minimum operating unit in InFrame. Note that, this design is based on our user-study finding that a properly selected $p$, which approximates the human eye resolution, can lead to minimal Phantom Array effect. For example, $p = 4$ is deemed a good choice for a screen with resolution 1920×1080 at typical viewing distance (1.2x the diagonal of the screen). Neighboring $s * s$ super Pixels in a data frame are treated as a *coding Block* (Block for short hereafter). For robustness, a Block carries one bit information. To cope with potential visible channel distortion, particularly due to rolling shutter effect, we further apply a simple parity-based error detection technique. We treat $m * m$ Blocks as a Group of Blocks (GOB for short). In each GOB, common error correction code such as RS code are applied. Further framing optimizations are permitted (*e.g.*, [4]); they complements our design.

• **Encoding and Multiplexing** A Block consisting of $s * s$ Pixels ($ps * ps$ pixels) carries one bit. We use a simple scheme to encode the bit. For bit 0, we set all Pixels to zero; whereas for bit 1, we encode it with a *chessboard* pattern. Therefore, the video content of an $ps * ps$ area will not change when multiplexing a 0 bit, but will add or subtract a same-sized patch with a chessboard pattern for bit 1. The chessboard pattern is simply generated by setting the Pixel at position $(i, j)$ to $\delta$, if $i + j$ is odd; or 0, otherwise. The amplitude $\delta$ is a tunable system parameter. Note that the pixel value of a multiplexed frame needs to be capped to the valid range *e.g.* [0, 255] for the 8-bit pixel. Therefore, for bright or dark areas, we locally adjust the amplitude for corresponding Blocks in two subsequent complementary frames. In our prototype, we form GOB from $2 * 2$ neighboring Blocks, and apply XOR-based parity checking, where the fourth Block bit is a parity bit generated by performing `xor` on the other three Blocks in the same GOB. Therefore, a frame can carry up to $w/s/2 \times h/s/2 \times 3$ bits.

Note that, in view of each multiplexed video frame, the introduced chessboard pattern is, in fact, the artificial noise added to the original video content. This artificial noise is further exploited for demultiplexing purposes. Moreover, it lets our coding scheme insensitive to the luminance and motion of the original video content.

• **Demultiplexing and Decoding** The process of demultiplexing is to evaluate the induced noises of the chessboard

pattern. During demultiplexing, for each GOB, we check the *induced noise levels* of the $2 * 2$ component Blocks. The process is as follows. We apply smoothing to that Block, subtract the smoothed content from the original content, and obtain the difference. We thus retrieve the noise level for the Block by summing up the absolute difference. The rationale of the process is that, the existence of the chessboard pattern (i.e., carrying bit 1) will surely lead to significant difference from the smoothed version. In contrast, the original video (i.e., carrying bit 0) is usually smoother and tends to yield much smaller difference from its smoothed version. Note that $\tau$ plays an important role in demultiplexing and data throughput. The larger the value $\tau$, the more likely we may subtract the smoothed content with more captured frames. On the other hand, the data throughput drops with the larger $\tau$. We evaluate its impact in §4. To work around high-texture areas in the original video content, we further remove the mean absolute difference. Finally, we apply a threshold $T$ to conclude the existence of significant induced noises.

A GOB is termed as an *available GOB* if all its component Blocks are decoded. For an available GOB, parity-based checking is then applied for the four Blocks in the GOB. If it fails the parity check, it is marked as an *erroneous GOB*. More sophisticated error correction codes can be applied for larger GOB. We leave this as part of the future work.

Note that, screen-camera communication differs INFRAME from existing studies on steganography and watermarking [16–23]. Their goal is to convey certain information over the original video with changes imperceptible to users. In contrast, INFRAME not only embeds data bits onto the video frames, but also ensures that the extra bits can be captured by the camera for data delivery. It thus seeks to maximally alter the original video frames to carry as much information as possible, as long as the video appears unaltered to the user via the flicker fusion at a high display rate.

## 4. IMPLEMENTATION & EVALUATION

The prototype of INFRAME is implemented in C# with 2000 lines of code. It consists of a sender and a receiver. The sender takes as its input an original video stream and a data frame stream, generates the multiplexed stream, and then plays back the video stream at precisely controlled frame rate (implemented using DirectX for experimental purposes). The receiver takes the captured frames as its input, detects the existence of chessboard patterns, and decodes the recovered data frames.

We evaluate INFRAME in two aspects. First, we conduct a user study to validate whether our design ensures normal video viewing without quality degradation or interference. Second, we assess the data communication performance (i.e., throughput) of INFRAME.

**Experimental Settings** We have used an Eizo FG2421 24' LCD monitor, which supports 120FPS frame rate and $1920\times1080$ spatial resolution. In our experiments, we set the brightness as 100%. We have used a Lumia 1020 device as the receiver to capture the display. The video-capturing resolution and frame rate are set to be $1280\times720$ and 30FPS, respectively. All experiments were conducted in typical indoor office settings at the capture distance of 50cm (about the desk width). We have used a pseudo-random data generator with a pre-set seed to generate the original data frames. Each data frame consists of 30*50 Blocks that are grouped into 15*25 GOBs. We have employed three different videos as the input ones, *i.e.*, a pure gray video, a pure dark gray video, and a normal sun-rising video clip. The pure colored video is adopted for its ease to detect any visual artifact. Their RGBs are (127, 127, 127) and (180,180,180), respectively. Typical multiplexed frames are illustrated in Figure 4.

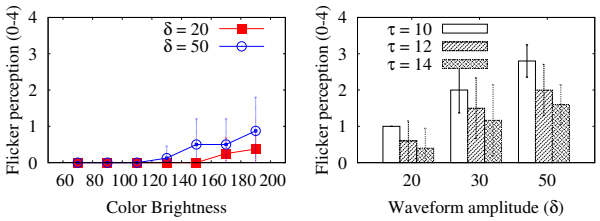



**Figure 6: The impact of data frame amplitude $\delta$ (left) and gradual-changing cycle $\tau$ (right) on flicker perception (average and standard deviation) in an 8-user study.**

**Subjective Assessment** INFRAME has several tunable parameters, including the temporal smoothing cycle $\tau$, the amplitude of data frame pixels $\delta$, *etc.*. We thus first evaluate the subjective quality, via user studies, to identify a good set of parameters so that the primary video quality is not impaired. We invited 8 participants, 3 female and 5 male, aged between 21 to 36, with half of them wearing glasses. Among the participants, there were a designer and a video expert, who are more sensitive to video quality. We showed original and multiplexed videos side by side, and asked them to rate the flicker (video quality change) with scores 0 to 4, where 0 indicates "no difference at all," and 1 to 4 signifies "almost unnoticeable," "merely noticeable," "evident flicker," and "strong flicker or artifact," respectively. In our user study, 0 and 1 denote satisfactory scores.

Figure 6(a) shows that INFRAME is quite tolerant to the "noisy" level of a data frame. The smaller the amplitude, the fewer the flickers. Even when the amplitude goes as large as 50, the resulting visual quality change is still mostly not perceivable. In all the tests, the average score is below 1 and most are 0 or 1. When the video turns brighter, the level of flickers or artifacts becomes stronger.

We further study the temporal smoothing cycle that enables transitions among different data frames in Figure 6(b). We see that longer cycles tend to reduce the perceived flickers. It takes a longer transition for the larger data frame amplitude to exhibit little impact on the resulting video quality. We also tested with other parameter settings, such as patterns, color, frame rate, Pixel size, and block size, in our user study. The results show that, our INFRAME design is

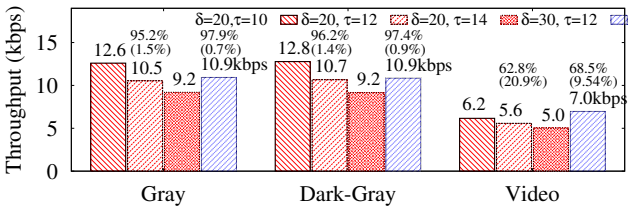



**Figure 7: The throughput of INFRAME with different input videos. The available GOB ratios (top) and the error rates (bottom) for $\tau = 12$ are given.**

able to safeguard clean video-viewing experience (*e.g.*, when $\delta \leq 20$, $\tau \geq 10$). We omit the details due to space limit.

**Screen-Camera Data Communication** Figure 7 show the achieved throughput, the ratio of available GOBs, and the error rates of GOBs in the preliminary performance assessment using four settings: $\delta = 20$, $\tau = 10$, 12 or 14 and $\delta = 30$, $\tau = 12$. A GOB is marked as available if all its component Blocks are correctly decoded. The available GOB ratios and error rates remain almost invariant with regards to varying $\tau$, only the results for $\tau = 12$ are shown. We see that INFRAME achieves about 9-13 kbps for pure colored videos. The performance is lower for real video clips (5–7 kbps), mainly due to the smaller number of available GOBs (about 62-68.5%) and the larger error rates (about 9.5–20.9%).

## 5. DISCUSSION AND ONGOING WORK

We have so far exploited the perception gap between the human vision system and the (camera) device to design INFRAME. Many interesting aspects remain to be explored.

**Applications** An appealing feature of INFRAME is that its screen-camera data communication is seamlessly associated with the primary screen-eye communication. It facilitates to automatically collect human activity contexts without incurring extra effort. For example, INFRAME can be used to carry additional details or side-information accompanying the primary video watching (*e.g.*, coupon links in the ad video, comments and highlights in live sports streaming.) We believe this new paradigm exhibits clear advantages over conventional solutions using two separate channels: one for video streaming and the other for data communication.

**Throughput** The achievable throughput does not sound impressive at the moment. Video capture is the limiting factor to higher throughput. In INFRAME, we have introduced several parameters, e.g., Block size ($s*s$), amplitude ($\delta$), and smoothing cycle ($\tau$). Each of them introduces a dimension for tradeoff, as far as the data rate of the secondary screen-camera channel is concerned. How to better balance the tradeoff, or even devise a more effective scheme to increase the screen-camera channel rate without interfering the primary screen-eye channel, is of great interest.

**Practical issues** We have implemented a strawman system for the feasibility study. Many practical issues remain to be addressed: (1) How to multiplex video and data frames on any display (TV, monitor, phone screen, etc.)? (2) How to make the multiplexed video be imperceptible under diverse video operations (e.g., the original video frame should be rendered when video viewing pauses)? (3) What are the associated computational cost and energy overhead?

## 6. RELATED WORK

**Watermarking and Steganography** Watermarking and steganography, which covertly embed data in signals (*e.g.*, images or videos), *seem* the most relevant techniques. Steganography seeks to hide the embedded information, so that the altered signal remains as close as possible to the original. To this end, only the LSB (least significant bit) of pixel values is manipulated in both deterministic [16–19] and stochastic manners [20–22]. [24] offers an early tutorial, while [25] presents an updated survey on steganography. Watermarking is mainly designed for authenticity verification or integrity check. Robustness is thus the top priority, whereas stealthiness is of less interest [23]. INFRAME differs from both in that its goal is to enable imperceptible data communication, rather than data hiding or authentication. Therefore, INFRAME has to address the screen-camera communication issues while they do not.

**Screen-Camera Communication** Visible screen-camera communication has recently become an active research area [1–4]. Our work differs from all such existing studies by enabling dual-mode, concurrent channels for data communication and video viewing. Most techniques proposed for the screen-camera communication (*e.g.*, advanced barcode design) should be applicable to INFRAME.

**Full-Frame Communication** Several studies seek to establish unobtrusive screen-camera communications [26–29]. [26, 27] apply watermark detection techniques upon consecutive frames, where the original image/frame is repeatedly displayed [26], or duplicated [27]. [28] proposes a hue-based barcode design to deliver static (tag) content only, and [29] conveys data bits by adjusting the hues of the image. Unlike INFRAME, they either work with static images or lack technical details. Moreover, INFRAME uses complementary frames to leverage flick fusion in video viewing.

## 7. CONCLUSION AND FUTURE WORK

In this paper, we present INFRAME, a system that enables simultaneous full-frame video-viewing experiences for users and screen-camera communications for devices, through multiplexing original video frames and data frames on the same display. The design is centered around the exploration of the flicker-fusion property of the human vision system and superior capability of modern displays and cameras. Preliminary results have confirmed the viability of INFRAME.

We believe INFRAME explores a new paradigm for screen-camera communications as it circumvents existing distractive and hard-to-use barcode design. As the capability gap between users and devices continues to expand, we envision the rise of opportunistic visual communications over the primary device-to-human channel.

## 8. REFERENCES


[1] S. D. Perli, N. Ahmed, and D. Katabi. Pixnet: interference-free wireless links using lcd-camera pairs. In *MobiCom*, 2010.

[2] T. Hao, R. Zhou, and G. Xing. Cobra: color barcode streaming for smartphone systems. In *MobiSys*, 2012.

[3] Q. P. Wenjun Hu, Hao Gu. Lightsync: Unsynchronized visual communication over screen-camera links. In *MobiCom*, 2013.

[4] A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, and G. Shen. Enhancing reliability to boost the throughput over screen-camera links. In *MobiCom*, 2014.

[5] D. A. Atchison, G. Smith, and G. Smith. *Optics of the human eye*. Butterworth-Heinemann Oxford, UK:, 2000.

[6] E. Simonson and J. Brožek. Flicker fusion frequency: background and applications. *Physiological reviews*, 1952.

[7] D. Kelly. Flicker. In *Visual psychophysics*, pages 273–302, 1972.

[8] J. Levinson. Fusion of complex flicker. *Science*, 130(3380):919–921, 1959.

[9] J. Z. Levinson. Flicker fusion phenomena rapid flicker is attenuated many times over by repeated temporal summation before it is" seen.". *Science*, 160(3823):21–28, 1968.

[10] D. G. Green. Sinusoidal flicker characteristics of the color-sensitive mechanisms of the eye. *Vision research*, 9(5):591–601, 1969.

[11] G. Brindley, J. Du Croz, and W. Rushton. The flicker fusion frequency of the blue-sensitive mechanism of colour vision. *The Journal of physiology*, 183(2):497–500, 1966.

[12] W. A. Hershberger and J. S. Jordan. The phantom array: a perisaccadic illusion of visual direction. *The Psychological Record*, 48(1):2, 2012.

[13] I. Vogels and I. Hernando. Effect of eye movements on perception of temporally modulated light. http://2012.experiencinglight.nl/doc/28.pdf.

[14] J. Roberts and A. Wilkins. Flicker can be perceived during saccades at frequencies in excess of 1 khz. *Lighting Research and Technology*, 45(1):124–132, 2013.

[15] N. Rajagopal, P. Lazik, and A. Rowe. Visual light landmarks for mobile devices. In *ISPN*, 2014.

[16] X. Liao, Q.-y. Wen, and J. Zhang. A steganographic method for digital images with four-pixel differencing and modified lsb substitution. *Journal of Visual Communication and Image Representation*, 22(1):1–8, 2011.

[17] H.-C. Wu, N.-I. Wu, C.-S. Tsai, and M.-S. Hwang. Image steganographic scheme based on pixel-value differencing and lsb replacement methods. *IEE Proceedings-Vision, Image and Signal Processing*, 152(5):611–615, 2005.

[18] R. Balaji and G. Naveen. Secure data transmission using video steganography. In *IEEE International Conference on Electro/Information Technology (EIT)*, pages 1–5, 2011.

[19] R. Kavitha and A. Murugan. Lossless steganography on avi file using swapping algorithm. In *Conference on Computational Intelligence and Multimedia Applications*, volume 4, pages 83–88, 2007.

[20] J. He, J. Huang, and G. Qiu. A new approach to estimating hidden message length in stochastic modulation steganography. In *Digital Watermarking*, pages 1–14. Springer, 2005.

[21] T. Sharp. An implementation of key-based digital signal steganography. In *Information hiding*, pages 13–26. Springer, 2001.

[22] J. Fridrich and M. Goljan. Digital image steganography using stochastic modulation. In *Electronic Imaging*, pages 191–202, 2003.

[23] Y. Zhang. Digital watermarking technology: A review. In *Future Computer and Communication (FCC)*, 2009.

[24] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn. Information hiding – a survey. *Proceedings of the IEEE*, 87(7):1062 –1078, July 1999.

[25] A. Cheddad, J. Condell, K. Curran, and P. McKevitt. Digital image steganography: Survey and analysis of current methods. *Signal Processing*, 90(3):727–752, 2010.

[26] W. Yuan, K. Dana, A. Ashok, M. Gruteser, and N. Mandayam. Dynamic and invisible messaging for visual mimo. In *IEEE Workshop on Applications of Computer Vision (WACV)*, 2012.

[27] R. Carvalho, C.-H. Chu, and L.-J. Chen. IVC: Imperceptible video communication. 2014. Demo.

[28] G. Woo, A. Lippman, and R. Raskar. Vrcodes: Unobtrusive and active visual codes for interaction by exploiting rolling shutter. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.

[29] T. Li, C. An, A. Campbell, and X. Zhoun. Hilight: Hiding bits in pixel translucency changes. In *VLCS*, 2014.