iMarker: Instant and True-to-scale AR with Invisible Markers

Chang Xiao
Adobe Research
United States

Ryan A. Rossi
Adobe Research
United States

Eunyee Koh
Adobe Research
United States

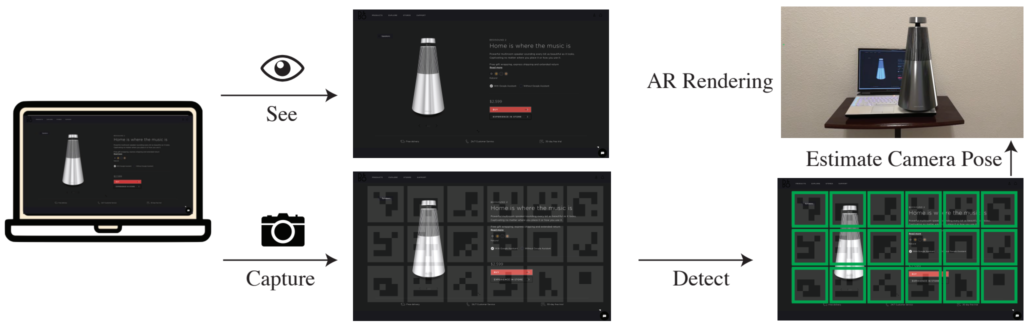

Figure 1: AR rendering via iMarker. Regular website content is delivered through screen-to-eye channel and iMarkers are delivered through screen-to-camera channel simultaneously. The iMarkers are visible through regular smartphone camera. Our system can easily detect the markers, track the camera pose, and lastly, render AR content.

ABSTRACT

Augmented Reality (AR) has been widely used in modern mobile devices for various applications. To achieve a stable and precise AR experience, mobile devices are equipped with various sensors (e.g., dual camera, LiDAR) to increase the robustness of camera tracking. Those sensors largely increased the cost of mobile devices and are usually not available on low-cost devices. We propose a novel AR system that leverage the advance of marker-based camera tracking to produce fast and true-to-scale AR rendering on any device with a single camera. Our method enables the computer monitor to be the host of AR markers, without taking up valuable screen space nor impacting the user experience. Unlike traditional marker-based methods, we utilize the difference between human vision and camera system, making AR markers to be invisible to human vision. We propose an efficient algorithm that allows the mobile device to detect those markers accurately and later recover the camera pose for AR rendering. Since the markers are invisible to human vision, we can embed them on any website and the user will not notice the existence of these markers. We also conduct extensive experiments that evaluate the efficacy of our method. The experimental results show that our method is faster and has a more accurate scale of the virtual objects compared to the state-of-the-art AR solution.

CCS CONCEPTS

• **Human-centered computing** → **Interaction devices**.

KEYWORDS

AR Markers, Hidden Screen-to-camera Communication, Augmented Reality

ACM Reference Format:
Chang Xiao, Ryan A. Rossi, and Eunyee Koh. 2022. iMarker: Instant and True-to-scale AR with Invisible Markers. In *UIST '22: The 35th Annual ACM Symposium on User Interface Software and Technology, October 29, 2022, Bend, OR.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Online shopping experiences have been largely enhanced by Augmented Reality (AR) techniques. Many shopping websites or Apps (e.g., IKEA) support AR view of their products on mobile devices. AR can help the customer to understand how the products fit into the desired environment before buying one, thus reducing the return rate. However, current AR solutions are usually based on markerless scene reconstruction, which requires the user to use their mobile device to scan the environment for a few seconds. The quality of scene reconstruction will determine the quality of virtual objects rendered in AR. In addition, the size of the rendered AR object may not be accurate due to lack of physical scale reference.

In this work, we propose a new marker-based AR technique that allows mobile devices to render true-to-scale virtual objects instantly near the computer monitor. An intuitive illustration of the proposed concept is provided in Fig. 1. In our framework, composite contents are produced for the display by multiplexing the webpage content frames (intended for human viewers) and the AR markers frames (intended for devices). These composite frames can be rendered to human eyes without affecting the viewing experience. Therefore, the user can browse the website as usual without ever noticing the embedded AR markers. In the meantime, the AR markers carried by the composite frames can be captured and decoded by a mobile phone camera to estimate the camera position and later on to render AR content. To enable the above functions, our method leverages the capability discrepancy and distinctive features of the human vision system and devices (display and camera). Most computer monitors have a refresh rate higher than 60Hz, while our human eyes have only a limited recognition capability lower than 60Hz [6, 8]. On the other hand, most modern mobile devices support high frame rate video capturing up to 240fps. Thus, if the monitor displays regular website content while overlaying flickering AR markers, the human eyes can only observe the original web content. At the same time, the mobile device can recognize the AR markers and use those as references for the 3D environment. Rendering virtual objects based on the detected AR markers then can be achieved through standard algorithms.

2 APPROACH

2.1 Overview

Our method consists of an encoding step and a decoding step. In the encoding step, we take a normal webpage as input, e.g., an online shopping webpage, and generate two webpages with AR markers array overlaid on the original content. These two images will later be alternatively displayed on the computer screen at a high frame rate (usually at 60Hz, depending on the monitor type). Since a normal human vision system can only recognize changes lower than 60fps [6, 8], the alternatively displayed AR markers will be invisible to humans. The idea of encode information in invisible markers has also been explored by many works in HCI field with different modalities [1, 5, 12]. On the other hand, most modern smartphones support recording video at a higher frame rate (e.g., 120fps, some models like iPhone 6 or later support 240fps). Thus, the change in AR markers is observable for smartphones. Using our computer vision algorithms, the centers of the embedded AR markers can be easily located. The detected centers will be later used for recovering camera poses by following the standard marker-based tracking procedure [4, 13]. Fig. 2 provides an overview of our encoding and decoding pipeline.

2.2 Encoding

The human vision system has a limited detection ability of time-variant fluctuation of light intensity [6, 8]. When the changes are above the *Critical Flicker Frequency* (CFF), the human vision system is not able to detect the changes, while only the averaged luminance is perceived. For example, if we have a white image and a black image displayed alternatively at a high frame rate, the human vision system will recognize it as a static gray image. According to the literature in vision research, CFF is affected by factors like background color, color contrast, and motion [6, 8]. It is believed that in most cases, CCF of human eyes is 40-50Hz [3, 6, 8]. Thus, flickering on a modern computer monitor (usually 60Hz) is unobtrusive to humans.

Given the characteristics of the human vision system, our goal is to generate two images where the averaged image under flickering is identical to the original image for human eyes; meanwhile, the two images should contain enough clear AR markers that are detectable by a high-frame-rate camera.

We generate the flickering image pair I^+_{ij} and I^-_{ij} as follows:

$$I^+_{ij} = I^0_{ij} + \alpha * \delta_{ij}, I^-_{ij} = I^0_{ij} - \alpha * \delta_{ij}, \tag{1}$$

where I^+ and I^- are the pair of images to be displayed alternatively at the monitor's refresh rate, ij are the pixel coordinate, I_{ij} denotes the luminance of image I at ij, I^0 is the original input image. $\delta_{ij} = 1$ if the AR marker is white at ij, otherwise $\delta_{ij} = 0$. We use ArUco [2] in our implementation as it is the most popular AR marker design. But it can be replaced by any other AR markers. α is a parameter for controlling the AR marker visibility. The higher α is used, the more visible the AR markers become in I^+ and I^-. The averaged luminance value of I^- and I^+ is equal to the original image I. According to existing studies [6, 8, 10, 11], the image resulted from flickering between I^- and I^+ should remain visually close to the original image under human eyes. Here we use the scale of 0 to 1 when representing the luminance.

2.3 Decoding

The goal of the decoding step is to detect the centers of the AR markers for each frame and use those detected centers to recover the camera position in absolute metric scale and render AR content. We first use a smartphone that can capture video under 120fps or higher. Most modern smartphones (e.g., iPhone 6 or later) support this feature. Initially, we investigated using the standard AR marker detection toolkit to extract the center of the markers frame by frame. However, we found that although each frame contains a visible portion of the markers, these markers are not fully detectable due to the low contrast of the original frames (see Fig. 2-e). Thus, we use adjacent frames to help detect the AR markers on a specific frame.

When detecting the markers on frame i, we use frames $i-2$, $i-1$ and $i+1$ and apply the feature-based image alignment technique [9] to align those image with respect to frame i. More specifically, we use ORB feature [7] which is efficient to compute and invariant to rotation and translation across each frame. Once the ORB features are computed and matched across each frames, we compute the homography matrix between frame $i-2/i-1/i+1$ and i, respectively. Then, we use OpenCV's `cv::warpPerspective` function to generate aligned image with respect to frame i. All frames are converted to grayscale before performing any process. Notably, this alignment is highly accurate with error that is negligible since we are recording at high frame rate and the motion between each frame is subtle. Let us denote the aligned frame as I_{i-2}, I_{i-1}, I_{i+1} and our target frame I_i. We derive I' as an transformed image of I as follows:

$$I' = (|I_{i-2} - I_{i-1}| + |I_i - I_{i+1}|)/2 \tag{2}$$

This process will generate a high contrast image I' on markers, because only the regions of where markers existed are different across these aligned frames. Next, given I', we can leverage OpenCV's AR marker detection function (i.e., `cv::aruco::detectMarkers`) to detect the centers of AR markers as well as its id on I' (see Fig. 2-f).

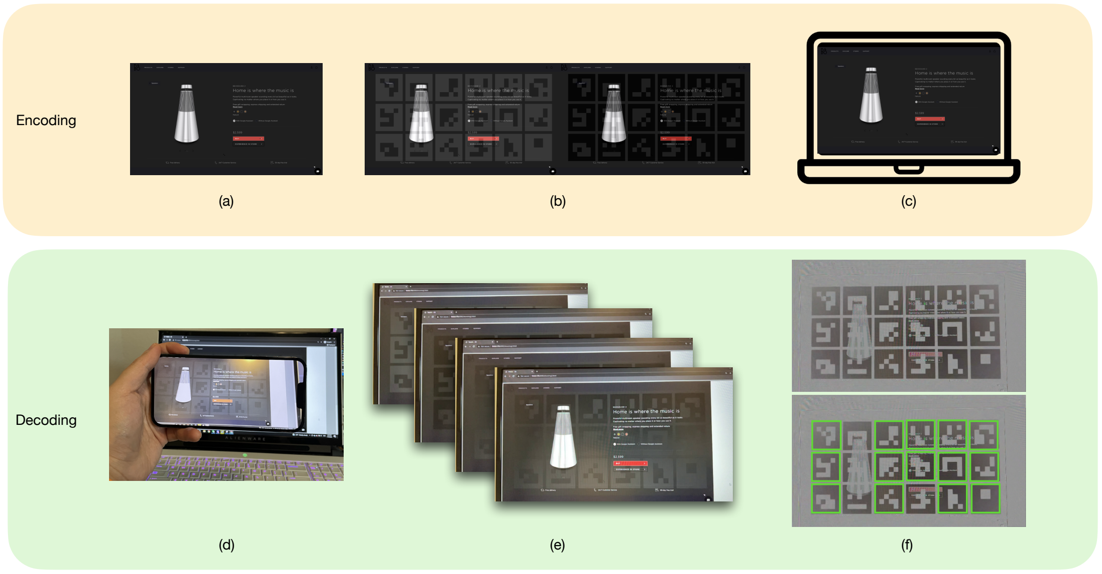

Figure 2: Overview of encoding and decoding steps. For encoding, we first take (a) the image of the original content (e.g., a webpage) as input. Then, we generate a pair of images which embeds the (b) I^+ and I^-, generated by the method described in Sec. 2.2. (c) A regular monitor is used to display the generated image alternatively, at its own refresh rate (e.g., 60Hz). The blended image appears identical to the original image under the human vision system. For decoding, (d) we use a smartphone to capture multiple frames at a high frame rate (e.g., 240fps). At each time step, we select (e) 4 continuous frames. By using the method described in Sec. 2.3, we synthesis (f) an enhanced image. We then apply the standard AR marker detection package to extract the center of those markers, which are later used to recover the camera position. Green boxes indicate successfully detected markers.

3 DISCUSSION AND CONCLUSION

Our systems also has several limitations. The first limitation is we need a monitor to host iMarker. This makes our system not compatible with AR applications that only use the device itself. Thus, we position our system as an auxiliary AR system that help the users to gain extra information in 3D (e.g., size) while watching content in a 2D display. Second, although in theory our method can work on objects with any size, in practice we found that our system will become unstable when dealing with objects much larger than a monitor. This is because if we want to see the full picture of a large object, we need to move the camera far away from the monitor. If the camera is too far from the monitor, the detection of iMarkers will be unstable as less pixels is contained in each iMarker. Last, our current system uses a single threshold α for the entire image. If the image has complex background color, using a single α may exceed the range of the intensity values at some part, which may generate observable spots for the users. In the future, we hope to explore adaptive way to set different α to different color while maintain high detection rate.

REFERENCES

[1] Mustafa Doga Dogan, Ahmad Taka, Michael Lu, Yunyi Zhu, Akshat Kumar, Aakar Gupta, and Stefanie Mueller. 2022. InfraredTags: Embedding Invisible AR Markers and Barcodes Using Low-Cost, Infrared-Based 3D Printing and Imaging Tools. In *CHI Conference on Human Factors in Computing Systems*. 1–12.

[2] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Rafael Medina-Carnicer. 2016. Generation of fiducial marker dictionaries using mixed integer linear programming. *Pattern recognition* 51 (2016), 481–491.

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

[4] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. 2009. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision* 81, 2 (2009), 155.

[5] Dingzeyu Li, Avinash S Nair, Shree K Nayar, and Changxi Zheng. 2017. Aircode: Unobtrusive physical tags for digital fabrication. In *Proceedings of the 30th annual ACM symposium on user interface software and technology*. 449–460.

[6] Robert L Myers. 2003. *Display interfaces: fundamentals and standards.* John Wiley & Sons.

[7] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*. Ieee, 2564–2571.

[8] Ernst Simonson and Josef Brozek. 1952. Flicker fusion frequency: background and applications. *Physiological reviews* 32, 3 (1952), 349–378.

[9] Richard Szeliski et al. 2007. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision* 2, 1 (2007), 1–104.

[10] Anran Wang, Zhuoran Li, Chunyi Peng, Guobin Shen, Gan Fang, and Bing Zeng. 2015. Inframe++ achieve simultaneous screen-human viewing and hidden screen-camera communication. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. 181–195.

[11] Anran Wang, Chunyi Peng, Ouyang Zhang, Guobin Shen, and Bing Zeng. 2014. Inframe: Multiflexing full-frame visible communication channel for humans and devices. In *proceedings of the 13th ACM Workshop on Hot Topics in Networks*. 1–7.

[12] Chang Xiao, Cheng Zhang, and Changxi Zheng. 2018. Fontcode: Embedding information in text documents using glyph perturbation. *ACM Transactions on Graphics (TOG)* 37, 2 (2018), 1–16.

[13] Zhengyou Zhang. 2000. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence* 22, 11 (2000), 1330–1334.