All that GLITTERs: Low-Power Spoof-Resilient Optical Markers for Augmented Reality

Rahul Anand Sharma
Carnegie Mellon University
rahulans@cmu.edu

Adwait Dongare
Carnegie Mellon University
adongare@cmu.edu

John Miller
Carnegie Mellon University
jmiller4@andrew.cmu.edu

Nicholas Wilkerson
Carnegie Mellon University
nwilkers@andrew.cmu.edu

Daniel Cohen
Carnegie Mellon University
danielco@andrew.cmu.edu

Vyas Sekar
Carnegie Mellon University
vsekar@andrew.cmu.edu

Prabal Dutta
UC Berkeley
prabal@berkeley.edu

Anthony Rowe
Carnegie Mellon University
agr@ece.cmu.edu

Abstract

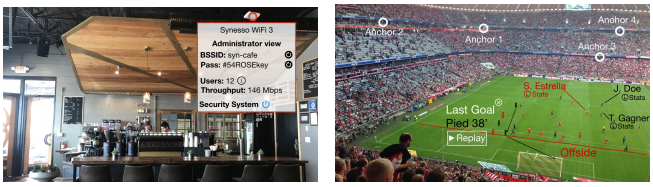
One of the major challenges faced by Augmented Reality (AR) systems is linking virtual content accurately on physical objects and locations. This problem is amplified for applications like mobile payment, device control or secure pairing that requires authentication. In this paper, we present an active LED tag system called GLITTER that uses a combination of Bluetooth Low-Energy (BLE) and modulated LEDs to anchor AR content with no *a priori* training or labeling of an environment. Unlike traditional optical markers that encode data spatially, each active optical marker encodes a tag's identifier by blinking over time, improving both the tag density and range compared to AR tags and QR codes.

We show that with a low-power BLE-enabled micro-controller and a single 5 mm LED, we are able to accurately link AR content from potentially hundreds of tags simultaneously on a standard mobile phone from as far as 30 meters. Expanding upon this, using active optical markers as a primitive, we show how a constellation of active optical markers can be used for full 3D pose estimation, which is required for many AR applications, using either a single LED on a planar surface or two or more arbitrarily positioned LEDs. Our design supports 108 unique codes in a single field of view with a detection latency of less than 400 *ms* even when held by hand.

1 Introduction

Augmented Reality (AR) systems overlay virtual content onto the physical environment through heads-up displays or mobile devices with pass-through cameras. One of the major challenges in AR systems is creating a reliable mechanism for registering virtual content on specific objects or at precise locations. The ideal solution to this challenge would be a mechanism for creating a *cyber-physical hyperlink* that can connect digital content to a physical asset or location without training or setup, in a secure and simple to deploy manner.

Consider a visit to a hypothetical sports arena where we would like to augment the physical space with intuitive AR interfaces. In the parking garage, an AR anchor attached to a parking meter could securely help users pay for parking. At the concession stand or at the cafe, as shown in Figure 1a, users could use these same AR anchors to obtain important information like WiFi credentials. Once inside the stadium, a constellation of AR anchors could annotate events in real-time with rich and interactive virtual content (e.g. previous goals, player statistics) as seen in Figure 1b. Such a constellation could also provide precise localization to help visitors navigate to their seats, to various sections of the arena and then safely back to their vehicle at the end of the event. The arena managers could use these same AR anchors to control the lighting and configure digital signage systems. Such a deployment would involve a large number of AR anchors that must be non-intrusive and easy to maintain while being simultaneously used by a large number of users for a variety of applications in a secure manner.



(a) Augmenting a device (b) Augmenting an environment

Figure 1: Concept sketches of virtual augmentations anchored in the physical world

There are a few critical requirements of an AR tag needed to support the above application scenario:

1. Compatible with existing off the shelf mobile devices: If we want tags to be broadly adopted, they need to be compatible with existing devices and AR frameworks that are optimized for displaying graphics (changing camera framerate, ISO and exposure) and not necessarily for image processing pipelines.
2. Operation in dynamic environment and lighting: Tags should be able to work across diverse environments (indoors or outdoors) and lighting conditions (bright or dark).
3. Demonstrative Identification[3, 24]: Anchors that support payment systems or IoT configurations should be resilient to spoofing attacks involving imitation and communication interception. Additionally, if a number of identical-looking devices are in front of a user, the user should be able to validate which device is linked to which digital identity. We use *demonstrative identification* to describe the user's ability to verify the identity of a tag that is within visual sight.

Given these baseline requirements, the following quantitative metrics are most crucial in defining the design space:

Accuracy Tags should provide precise locations and operate across distances spanning tens of meters.

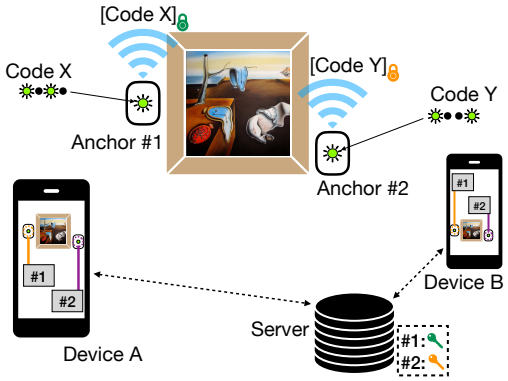


Figure 2: GLITTER System Overview. Anchors modulate LEDs to transmit codes that are detected by mobile phones and used to overlay AR content onto physical objects.

Detection latency Tags need to be detected quickly enough to support human interaction. We also see that latency directly impacts robustness to hand motion.

Scalability Many tags should be able to concurrently support a large number of users.

SWaP-C (Size, Weight, Power and Cost) Tags should be small, low-cost and operate for long periods.

The most widely used approach for anchoring AR content is through visual markers like AR and AprilTags [9, 26]. Since these tags are passive (do not change), they are easily cloneable and do not provide strong authentication guarantees. The other major class of approaches for registering AR, used by commercial headsets and mobile AR, is based on scanning an environment using visual features [30, 38] or depth cameras [21, 25]. These approaches struggle in the presence of dynamic objects and lighting. They are also not able to track similar-looking objects in the space without unique visual markers. Specialized localization hardware can use optical trackers [33], arrays of RFID readers [23], mmWave radios [22], Time-of-Flight (ToF) and Bluetooth Direction Finding [4] and ToF Ultra-Wide Band (UWB) [8] radios to localize devices within a scene. These systems vary in terms of accuracy, are difficult to provision, and often require expensive additional hardware that is not currently accessible on common mobile platforms.

While visual markers such as AprilTags seem like a promising solution to many of our requirements, they fail to be spoof resistant, have limited range and suffer poor performance under dynamic lighting. To address these shortcomings, we leverage the following insights (Section 3):

A. Active Visual Markers: Any static marker is vulnerable to spoofing, so by making visual tags active (dynamic) we can make them non-spoofable and support Demonstrative Identification.

B. Space vs. Time Trade Off: Commodity AR devices have a limited resolution that bounds the detection range of visual markers. If our markers are active, instead of encoding the data spatially, we can encode data temporally to achieve a longer range and better performance under dynamic lighting.

C. Hybrid BLE + Visual Anchoring: Due to tight latency constraints, a noisy visual channel and limitations imposed by commodity phones, it's impractical to use a device's camera as a traditional communication channel (preamble + data + CRC) for Demonstrative Identification. To this end, we develop a hybrid communication scheme wherein a tag can transmit its blinking pattern over BLE, and a client needs only perform a significantly shorter search for this pattern in the scene. This practical design is backed by secure crypto primitives and guarantees Demonstrative Identification under reasonable attacker assumptions.

Building on these insights, we present GLITTER (Glowing Light Identification Tags That Enhance Reality), a system that uses a coded blinking LED in conjunction with Bluetooth Low Energy (BLE) [10] to discover and localize AR content using cameras. We can also use a constellation of these tags to estimate full 3D pose. By blinking the LED, we can change its code as part of a challenge response protocol used for authentication. The time-varying nature of the channel can dramatically reduce the size of the tag, making something as small as a single pixel detectable. Figure 3 shows the size vs detection range comparison of GLITTER with AprilTags.

In order to decrease latency and increase robustness, we leverage BLE as the primary communication channel for data and only use the light channel as a way to associate tags within the camera's view. Even though LEDs are pervasive on devices and people tend to ignore them, it is important to note that in most applications, the markers are triggered as needed and on-demand to save power.

Figure 2 shows the overall system architecture with two mobile phones detecting two active optical markers. Each tag acts like a standard BLE beacon in advertising mode, except that clients can reply to the broadcast (in a scalable manner described in Section 5) in order to trigger a single coded LED flashing sequence. These codes are detected across multiple frames and used to determine the identity and position of the device. In cases where the authenticity of the device is important, the code is protected and can be verified by a trusted authority using a Message Authentication Code (MAC). In our reference implementation, we are able to simultaneously detect 108 active optical markers within 400ms (6 bits over 12×2 frames) at ARKit's highest resolution of 1920x1440 while running at 60 fps. The system can process at up to 240 fps without ARKit support for applications that demand even lower latencies.

We implement GLITTER as an open-source iOS framework with a demonstration project for Unity as well as integrated into a modified version of Mozilla's XR Viewer (an experimental mixed reality web browser). When using XR Viewer, GLITTER uses the BLE UUID to download a description of any nearby constellations that when detected are used to warp the camera location of the AR frame. As an example, we show in a theatre environment that GLITTER provides a median pose estimation error of 7.47° and a location error of 0.34 m over a distance of 30 meters when providing global-world anchoring.

Contributions and Roadmap: In summary, this paper makes the following contributions:

1. An approach for accurately identifying many active optical markers in an AR video sequence (given limited control over ISO, exposure and frame rate) in the presence of camera shake.
2. A protocol for Demonstrative Identification that is resilient to spoofing.
3. A low power network protocol for advertising over BLE and blinking codes to multiple users.
4. An evaluation of GLITTER's pose estimation used to render AR content.
5. An iOS framework compatible with Unity as well as Mozilla's XRViewer mixed reality web browser that can display webXR pages anchored in global coordinates.

2 Related Work

Work related to AR registration can be roughly broken down into four major categories: (1) computer vision approaches, (2) fiducial markers, (3) specialized localization systems and (4) visible light communication. Table 1 shows a comparison of many of the most common and promising approaches described below.

Computer Vision: Headsets [21, 25, 35] and recent mobile AR platforms like ARKit [2] and ARCore [11] use visual and/or depth features to relocalize given a previously scanned space. These systems use various forms of Simultaneous Localization and Mapping (SLAM) [32] which have the advantage of not needing any tags. SLAM requires movement in the space before a location can be determined (increasing acquisition latency) and doesn't work well in low-feature environments or when the scene changes. Active optical markers could easily complement these approaches when available, to dramatically improve acquisition time and robustness.

Fiducial Markers: As a technique to improve robustness, there has been extensive work in coded visual tags and optical markers, like AR Tags tags [9], ARToolKit [36], AprilTags [26] and QR codes [5]. QR codes are designed primarily for encoding data like web links, while AprilTags are designed primarily for estimating distance and orientation. While cheap and effective, these approaches require large and often obtrusive tags that are easy to copy and spoof [17, 34]. Commercial options like Vueforia [38] can learn features from arbitrary images that later act as codes. These can be more easily concealed but are limited in number and tend to be less robust compared to tags specifically designed for positioning.

Specialized Localization Solutions: Recent commercial VR systems use either beacons or trackers to localize headsets. The HTC Vive [14] uses a sweeping IR laser in its lighthouse system that can detect horizontal and vertical angle based on the arrival time of the sweeping signal on an IR detector. While extremely accurate, this requires powered beacons and a custom receiver. The Oculus Rift headset uses a technique that is similar to GLITTER where a constellation of IR LEDs on the headset are detected by a fixed IR camera to determine pose information. Each LED in the constellation blinks a 10-bit code that is captured and decoded by a 60Hz IR camera. This system requires tight (wired) time synchronization between the blinking IR LED and the camera and does not focus on being robust to visual background noise if it was not in the IR light spectrum.

Approaches like motion capture systems [33] or RF based systems like GPS [27], 3D RFID tracking [23] and UWB localization [8] have shown increasing potential in supporting AR applications. If the pose of the tag and the device can be directly measured from an external system, the relative location can be computed on an image. Unfortunately, these approaches require expensive infrastructure and additional hardware and are not robust in clutter or high multi-path environments or over large distances.

Visible Light Communication: The Visible Light Communication (VLC) community has produced a large body of work that focuses on using light as a communication channel. Most of the previous work in this space requires specialized hardware for transmitting up to 100MHz [19, 31] with trichromatic (RGB) LEDs, or up to 20MHz with the more ubiquitous phosphorescent white LEDs [12, 13]. Much of this work is captured in the 2006 IEEE VLC standard known as 802.15.7 [15]. Within the VLC space, the closest related work is in the area of camera communication, where researchers have investigated using LEDs to communicate with standard cameras. Danakis et. al. [7] exploit the rolling shutter effect of a smartphone's CMOS camera to capture On-Off Key (OOK) modulated data from LEDs. The authors generate data sequences using Manchester encoding, resulting in multiple symbols being captured per frame. VRCodes (NewsFlash) [37] take advantage of the rolling shutter effect to decode visual tags displayed on LCDs. In all cases, the focus is on transmitting data and not on localization or enabling low-power tags for AR content. The feasibility of these techniques may change over time as cameras shift towards global shutters.

Other similar approaches like [28] and [20] use the rolling shutter effect to identify and localize frequency modulated LEDs for localization. These systems are designed for large powered LEDs that provide comparatively coarse-grained localization. Rolling shutter detectors need large enough targets to detect stripping across the image which severely limits range. In order to detect high-frequency components in rolling shutter camera systems, the exposure of the camera needs to be set very low to avoid pixel saturation, which results in a extremely dark image that can't be used for AR.

CamCom [29] is one of the few VLC techniques that works given a global shutter camera by using under-sampled frequency shift OOK (UFSOOK). UFSOOK works by encoding data at frequencies that are harmonics of the frame rate, and decoding data by processing the subsampled aliased frequencies. CamCom has the goal of being a flicker-free communication channel whose data transfer is unnoticeable to humans. For this reason, they sacrifice range and latency, making it a poor choice for our particular use-case as it is extremely sensitive to camera shake. They also don't explore the ramifications of demodulating tags in the AR context where you may not have control over camera settings.

The most similar work to GLITTER is [1], which uses OOK modulated LEDs to anchor AR interfaces in 2D. The anchors presented in this work use a computer vision approach based on template matching to locate bright spots across an image pyramid. LEDs are OOK coded with a preamble, data payload and checksum at 120 Hz. Each candidate point from the image tracking step is then validated by attempting to decode a full packet. Since this approach transmits an entire data packet, it requires a higher camera frequency that is not yet compatible with AR Kit/Core. The authors report a 10% BER at distances of 5 to 10 meters, whereas our implementation can operate up to 30 m with an even lower BER of 5%. Their approach is entirely one-way light communication, which simplifies adoption but does not provide protection against spoofing (replay attacks) and takes an average 1.5 packet lengths to ensure detection. In our proposed approach, the light can take any shape (useful for entertainment applications), while this work requires that the light be a point source. One could imagine a hybrid approach that mixes shapes with blinking lights to identify and decode tags. Finally, this work only looks at detection on a 2D image and does not look at capturing camera pose which would stress the technique's ability to accurately decode multiple tags simultaneously.

3 System Overview

After iterating through the various AR anchoring options in Table 1, we can see that while visual markers such as AprilTags satisfy many of our system requirements, they fail to be spoof resistant, have limited range, and suffer poor performance under dynamic lighting. We assume AprilTags as our strawman solution and go over solutions to their limitations.

Any static tag is vulnerable to spoofing because an attacker can observe the tag and learn its pattern. AprilTags and QR codes are extremely vulnerable to spoofing, as anyone can print a similar

System	Anti- Spoofing	Robustness		Compatibility with COTS devices	Accuracy	Concurrent Users	SWaP-C		Latency
		Environment	Lighting				Size	Power	
April Tags/QR Codes	No	No	No	Yes	High	High	Large	No	Low
GPS	Yes	No	Yes	Yes	Low	High	Small	High	High
Vuforia	No	Yes	Yes	Yes	High	High	Large	Low	Low
RFID	No	limited	Yes	No	High	High	Small	No	Low
VLC (rolling shutter)	No	Yes	Yes	No	High	High	Small	Low	Low
Light Anchors	No	Yes	Low	Yes	High	High	Small	High	Low
BLE	No	Yes	Yes	Yes	Low	High	Small	Low	Low
UWB	No	Yes	Yes	limited	Medium	Medium	Small	Medium	Low
Optical/Tracking	No	Yes	Yes	No	High	Low	Large	No	Low
GLITTER	**Yes**	**Yes**	**Yes**	**Yes**	**High**	**High**	**Small**	**Low**	**Low**

Table 1: Comparison of AR Anchoring techniques

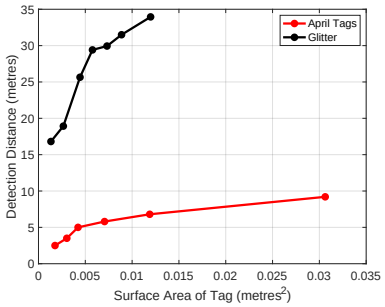


Figure 3: Tag size vs. detection range of GLITTER vs. AprilTags. Both experiments were performed by a user holding the phone in hand.

tag and paste the new tag over the previous one. Moreover, they could easily replicate the tag and place it in a new location. This is a limitation of the tag being passive and can be remediated by making it active (dynamic). This leads to our first insight:

A. To make tags support Demonstrative Identification they need to be active instead of passive.

Another challenge with our strawman solution is the detection range. Commodity phones have cameras with limited resolution, which means we are limited in range with spatial coding. Figure 3 shows the size of an AprilTag required to support a sport scenario. Tags of these sizes are impractical and obtrusive. This leads to our second insight:

B. By encoding the data temporally instead of spatially, we can achieve longer range and better performance under dynamic lighting.

Since we can encode the data temporally, our tags can be very small (even a single pixel). With a sufficiently long code, we can develop detectors that can accurately pick the source of the correct blinking pattern while rejecting other sources of light. Blinking a pattern on an LED and accurately detecting it is still a challenging spoofing problem due to the following constraints:

- No time synchronization between the tag and client.
- Limiting the system to the use of one-way (broadcast) communication between the tag and client. This is critical if we want to support a scenario where hundreds of clients are trying to communicate with a single tag (like at a sporting event).
- Tags may not have network connectivity after being deployed.

Let's say a tag can blink an unknown time-varying sequence X_t, and a client can observe and detect this sequence. This system could provide all the necessary security guarantees. A client can decode this data and verify the authenticity of the tag via a trusted third party. The client can then estimate their relative position using the detected location of the tag. Although this system provides all the necessary security guarantees, it is impractical due to the following concerns:

- Our channel is extremely noisy due to flickering lights, hand motion, etc. We need at least 10 data bits to get sufficiently low false positive rates (Figure 11b).
- We can only transmit 10 bits of information (including preamble, data and CRC) if we want to support handheld use and interactive latency.

This makes it impractical to use the blinking light channel for data transfer, but we can instead use it to detect shorter codes. If the client were to know the code *a priori*, it could perform a search for this pattern in its field of view, without the need for preambles.

All mobile phones and many embedded platforms have low-power radios like Bluetooth Low-Energy (BLE). A tag could broadcast its $f(key)$ over BLE and a client could then verify the tag authenticity with a trusted third party. Although this system can prove the authenticity of the tag, it doesn't provide the location of tag. Also, this system is vulnerable to a man-in-the-middle attack, as an attacker can easily mimic this $f(key)$ to trick a client. This leads to our third insight:

C. A hybrid scheme using both visible light and BLE communication can provide a practical design that satisfies the necessary security guarantees.

A tag can now transmit $f_1(key, t)$ over BLE and correspondingly blinks a sequence $f_2(key,t)$. Now, a client communicates this $f_1(key, t)$ to a trusted third party and gets the $f_2(key, t)$. It can then search for the blink sequence $f_2(key,t)$ and extract its relative position.

Our system, GLITTER, consists of two components: a.) A Visible Light channel (blinking LED) for anchoring and b.) A Hybrid Communication protocol for demonstrative identification. In the following sections, we go over the design of both of these channels in more detail.

4 Visible Light Anchoring

In this section, we discuss the design of our visible anchoring approach, starting with the LED transmitter and ending with the demodulator design that runs on images captured by a camera. This discussion includes various challenges that emerge from camera systems that often do not expose low-level controls over camera parameters like focus and ISO.

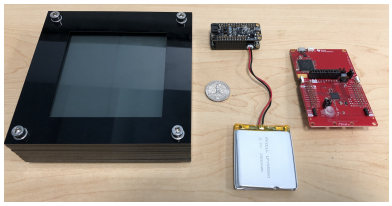


Figure 4: The family of platforms implementing GLITTER. (left to right) Large nRF32 GLITTER box for theater applications, Adafruit nRF52 Feather and TI CC2640 Launchpad.

4.1 Transmitter Design

GLITTER transmits data by blinking an LED using a duty-cycled On-Off Keying (OOK) scheme. The use of cameras as receivers impacts LED selection, symbol rate and the underlying transmit duty-cycle.

*4.1.1 LED Selection*One possible approach would be to use color as a feature for detecting and coding. In our experiments, we saw that bright LEDs tend to saturate camera sensors at relatively close ranges, making color a poor feature for coding. Since most camera sensors have more green receptors (driven by the nature of the human eye), green LED tends to provide the best sensitivity and hence the longest range given the lowest power. Smartphone cameras heavily filter light that is outside the visible spectrum, which eliminates the use of infrared or ultraviolet LEDs. A brighter LED has a higher signal power and can be recorded at much longer distances, but is expensive, power-hungry and might irritate users. Instead, a larger or diffused LED is preferred as it provides resilience to camera shake by increasing the detectable surface area. Interestingly, since our algorithm is dependent on changes in intensity, we see a higher signal-to-noise ratio with a dimmer LED that has a darker color when off, as opposed to LEDs that have white or clear color when off. To increase contrast, we use either a 5mm neutral density diffuser or a neural density gel film in front of an LED for our experiments.

*4.1.2 Symbol rate*Current AR frameworks such as ARKit have a maximum camera frame rate of 60 frames per second (fps), i.e. every pixel is sampled at a rate of 60 Hz. Although we could implement our own custom frame capture pipeline, we would lose the features provided by the AR frameworks such as continuous tracking using VIO (visual inertial odometry) and all of the existing software tools.

As cameras and supported AR frameworks become capable of higher frame rates, it will become possible to upgrade the symbol rate to $(FPS/2)$ Hz.

*4.1.3 Modulation Scheme*Most embedded systems only have binary control over LEDs driven by GPIO (General-purpose input/output). It is possible to change the observed intensity of the LED through pulse-width modulation. Unfortunately, even a small amount of exposure quickly saturates most camera sensors at shorter distances, limiting the system to OOK as opposed to amplitude based techniques.

There is also a timing offset between the start of a symbol transmission on the anchor and the start of the frame capture on the camera due, since the transmitter and receiver are not synchronized. With rolling shutter cameras, which sample different rows of pixels at different times, this adds an additional timing offset determined by the location of the LED on the image. The consequence of any timing offset, as illustrated in Figure 9, is that a symbol spreads over multiple samples. In the worst case, a symbol can spread over three frames (or even four frames, if there are frequency offsets between the anchor and device clocks), which is very inefficient. By limiting the On time of the LED to $t_{\text{on}} = 1/2 \times 1/FPS$, we can limit this spread to two frames while maximizing the amount of energy in the symbol.

As a result of these limitations, our modulation scheme consists of two symbols: (1) an Off symbol where the LED is not transmitting, and (2) an On symbol where the LED is transmitting for the first t_{on} seconds.

*4.1.4 Message Length*The number of symbols in a GLITTER message must be large enough to be reliably detected while rejecting false positives caused by other noise sources in the environment. For example, the pattern caused by a single symbol could naturally occur due to motion or other blinking LEDs, but the likelihood of a pattern of 12 symbols matching the pixel's binary values is exponentially smaller. The drawbacks of longer messages are that they consume more power and take more time. Longer messages are susceptible to errors caused by motion, since parts of the message may be captured by completely different pixels. We recommend a message length of 12 symbols as it is the smallest message size that does not cause significant false positives in the multiple environments we tested. We go into more detail about this selection criterion in the experimental evaluation section.

*4.1.5 Balancing*GLITTER only outputs signals when the LED is On, so a packet's detectability is dependent on the number of Ons contained in a packet. A balanced encoding scheme equalizes the number of On and Off symbols (ones and zeros), guaranteeing that every message has a similar chance of being detected regardless of its data contents and every message consumes the same amount of energy. It also guarantees that there are enough On and Off symbols in close proximity for the bit detector described in Section 4.2.3.

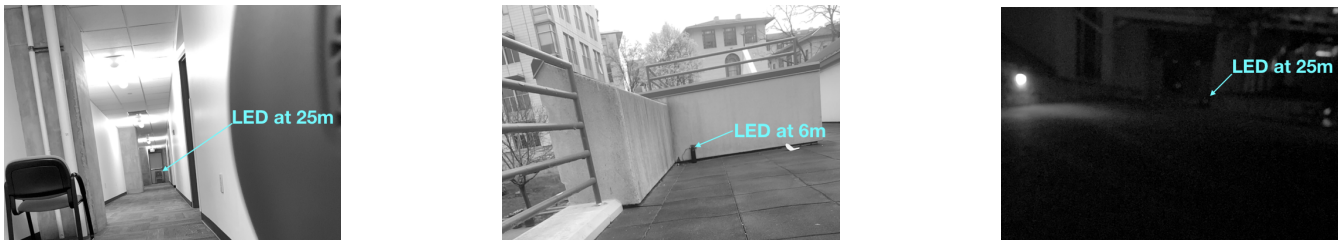
There are C_n^{2n} ways to create a balanced word of size $2n$ (choose the n ones in a $2n$ size word). Larger words are more efficient (encode more information for a given size), while smaller words contain ones and zeros in closer proximity. Given the message size of 12 symbols from Section 4.1.4, we choose to use three words made of four symbols (from a choice of 2, 3, 4, 6 or 12 symbol words). Words made of four symbols provide good efficiency for our data size ($\log_2 6/4 = 0.65$), have a good distribution of ones and zeros throughout the message and are simple enough to be implemented as a look-up table. If larger word sizes are desired for higher speed cameras, one could implement Knuth's efficient balanced encoder and decoder [18].

Each word can represent numbers from 1 to $C_2^4 = 6$: 0011, 0101, 0110, 1001, 1010, 1100. We constrain our first word to start with a 1, which halves the available choices. Each message can encode a number from 0 to $108-1$ (i.e. 0 to $3\times6\times6-1$) by representing it in base 6, mapping each digit to a word and then appending a 1 at the start.

*4.1.6 Low-Power Operation*Figure 6 shows a typical power trace of a TI CC2640 Launchpad (prototype platform for the GLITTER active optical markers) sending BLE advertisements and blinking its on-board LED. We estimate the energy consumption as

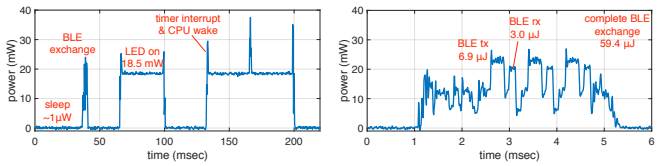
$$\begin{aligned} E_{\text{msg}} &= E_{\text{BLE}} + E_{\text{MCU}} + E_{LED} \\ &= E_{\text{BLE}} + 2n \times E_{\text{int}} + n \times P_{\text{LED}} \times t_{\text{ON}} \end{aligned}$$

$E_{\text{msg}} = 2.3\mu J$ is the energy consumed by the entire GLITTER transaction.



(a) Indoor/Office (b) Outside (c) Out of Focus Dark

Figure 5: Snapshots of On LEDs in varying lighting conditions



(a) Complete GLITTER message (b) Detailed view of the BLE transaction

Figure 6: Power trace of a GLITTER anchor

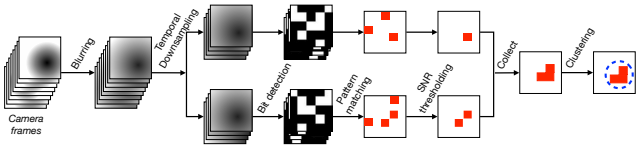


Figure 7: Decoder Block Diagram

$E_{\text{BLE}} = 59.4\mu J$ is the energy of the BLE advertisement shown in Figure 6b.

$E_{\text{int}} = 16.4\mu J$ is the energy of the timer interrupt, which occurs twice per symbol and is responsible for waking the processor and changing the LED state.

$E_{LED} = n \times E_{symbol}$. Each LED symbol consumes $P_{\text{LED}} \times t_{\text{ON}} = 18.5 \times 1000/60 = 308.3\mu J$.

The energy consumption of the blinking LED (per transaction) is significantly greater than the energy consumption of any other components on the anchor. Transmitting even a single symbol costs 5× more energy than a complete BLE exchange. Transmitting a balanced 12 symbol message costs 30× as much energy as a BLE exchange. A device running on two AAA 1000 mAh alkaline batteries blinking continuously would run out of energy in approximately half a month. The same anchor transmitting only two BLE advertisements every second without blinking its LED could last as long as 21 months.

Because of this energy consumption disparity, it is important to only blink when there is an AR device in proximity requiring its services. A BLE advertisement based protocol is described in Section 5 which efficiently implements this functionality even in the presence of many AR devices and active optical markers.

Energy harvesting devices can also be active optical markers as long as they can gather and store sufficient energy ($2.3\mu J$ plus losses). Based on our measured power traces, an energy harvesting platform like [6] can use its 10 cm^2 solar panels to recharge over approximately one minute and send one GLITTER message. In case there is no AR device requiring its service, we could use the above protocol to only send BLE advertisement packets when requested by a mobile device through a BLE response which would be immediately followed by an LED transmission.

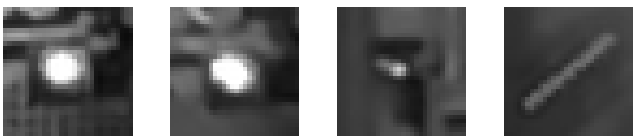
4.2 Detector Design

The detector consists of seven ordered stages as described below and shown in Figure 7. The system first blurs the image to combat camera shake, splits the streams into odd and even frames to account for frame timing offsets, digitizes bits, matches codes, thresholds detected regions based on SNR and then performs image clustering to find the center of the LED. Each step is detailed below.

4.2.1 Blurring Figure 8 shows example frame captures of an LED while the camera is being held by hand. Even over a short exposure time of 16.6 msec, the LED moves across multiple pixels due to camera shake, particularly at longer distances. This is a challenge for the detector, as different parts of the message may be located across multiple pixel locations and the shape may not even resemble the shape of the original LED.

Blurring spreads the LED energy over more pixels, which allows for a greater level of hand motion over the duration of a packet. Hand motion can cause false detections when sharp edges in the image oscillate between adjacent pixels. Blurring also helps remove false detections by performing a low pass filter so that only significant high frequency components pass through. We use a standard 2D Gaussian blur kernel and perform a 2D convolution across the frame to blur it. Gaussian blurring can be controlled by a parameter σ which spreads the signal over many neighboring pixels, which in turn reduces the signal-to-noise ratio (SNR) and reduces range.

4.2.2 Temporal Downsampling As discussed in Section 4.1.2, the symbol rate is limited to 30 Hz despite having a 60 FPS camera. However, timing offsets between the start of the pixel capture and the start of the symbol will cause a symbol to spread across into multiple frames. This effect is illustrated in Figure 9. At shorter distances, even a small amount of exposure could completely saturate the pixel. Techniques like equalization which are often used



(a) steady at 1 m (b) shaky at 1 m (c) steady at 10 m (d) shaky at 10 m

Figure 8: 30×30 pixel image of an LED.

in radios do not function well due to rapid saturation of the pixel in the presence of an LED.

To alleviate this problem, we downsample the camera stream into two parallel streams of 30 FPS and process them independently. All incoming frames are placed into one of two streams in an alternating manner (odd/even) and processed. The stream with the best signal match is selected as it best captures the phase of the signal.

4.2.3 Bit Detection Based on the findings in Section 8.3, we determine that our communication channel does not behave like an Additive White Gaussian Noise (AWGN) channel. Thus, techniques designed for AWGN-like channels, e.g. matched filtering for preambles, do not work well and are very sensitive to false detections.

To alleviate false detections, we binarize all the frames before searching for codes. We use the average of the min and max pixel values in the last k frames (in our case $k=6$) as our threshold given by:

$$\text{threshold}_i = \frac{\max(I_i,\ldots,I_{i-k})+\min(I_i,\ldots,I_{i-k})}{2}$$

It is necessary to find the threshold in small windows, since the changing camera focus and motion can drastically change the observed values of the On and Off symbols over time. Processing a stream of binary frames is also computationally faster.

4.2.4 Pattern Matching Next, we use pattern matching to find the sequence of bits in the stream of binary frames. We take the temporal sequence of decoded bits at a particular pixel location and XOR it with the desired sequence, hence a zero result indicates the pattern was found.

Figure 11b shows that searching for a longer pattern reduces the chances of false positives. However, it also increases the length of the message, which makes it more expensive in energy and more susceptible to errors caused by motion. If we were to transfer data over the light channel, we would have to interleave data with a known sequence of bits. From the above figure, this known sequence would have to be 12 bits or larger to avoid false positives from random fluctuations across frames. This makes the entire message longer than 400 msec and very susceptible errors from motion. Thus, we decide to use GLITTER as a detector where instead of decoding data, we search for a known sequence sent over BLE.

The supporting BLE channel is more efficient and reliable for data transfer while the LED is better for localization.

4.2.5 SNR Threshold Based on our observations from Section 8.2, our intuition is that the power of the LED causing a pattern match is much higher than matches caused by false positives. We later evaluate this intuition through receiver-operating characteristic (ROC) curves in Figure 15. Due to the nature of our channel and our use of OOK modulation, we only observe the signal while the On symbol is transmitted. If we record the signal during the transmission of an Off symbol, we only see noise. Thus we can define signal-to-noise ratio as follows:

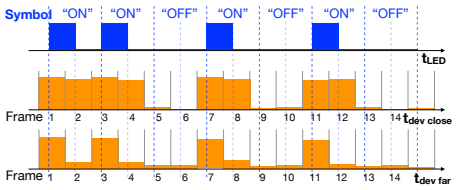


Figure 9: Symbol spreading across frames due to timing offset

$$SNR = \frac{P_{\text{signal}}}{P_{\text{noise}}} = \frac{I_{\text{on}}}{I_{\text{off}}}$$

The SNR for the actual message is calculated by taking the ratio of the average power of all the On symbols and the average power of all the Off symbols. If the SNR of a detection is lower than the SNR threshold, this detection is discarded. We compute the SNR threshold after pattern matching since we must know the positions of all the On and Off symbols to compute the ratio.

4.2.6 Clustering Typically, a number of pixels match the pattern and cross the SNR threshold, seen as a cluster of red pixels in Figure 10a. However, to anchor an AR object in reality, we must collect and reduce these multiple detections into a single point (assuming there is only a single anchor blinking a particular code in sight).

We use the centroid method (averaging the X and Y pixel coordinates) to find the center of the cluster. We also compute the variance along the two dimensions and compare the sum to a variance threshold. We set the threshold such that it eliminates sparse detections (primarily caused by spurious sources) while still accommodating the entire LED at close distances. We choose a variance threshold of 150 empirically (approximately 1/10 the frame dimension). The evaluation of more advanced clustering algorithms is left as future work.

5 Hybrid Communication Protocol

Bluetooth Low Energy (BLE) is primarily responsible for communicating the identity of the tag and informing the GLITTER detector about the code to find in its field of view. As shown in Section 4.1.6, LED transmissions consume 30× more power than BLE transmissions. To conserve battery energy, the tag must blink only on demand, which would have to be triggered using BLE communication.

A system where many AR devices need to maintain an active connection to the tag is not scalable. We are thus limited to the use of BLE advertisements (connectable and non-connectable) which have a small payload size of less than 30 bytes. Additionally, information communicated over BLE must be encrypted and verifiable to prevent spoofing attacks. As the tags must be low-cost and low-power, we cannot assume access to a network connection or guarantee any long-term time synchronization.

The rest of this section describes how we can use a hybrid of BLE communication and visual anchoring to satisfy the above requirements and provide demonstrative identification.

5.1 Demonstrative Identification for Security

Demonstrative identification is the user's ability to verify the identity of a tag in line-of-sight. Our system is secure against attacks by the following adversaries:

1. Adversary with duplicate LEDs: The attacker has created a malicious LED to trick the user. We can protect against these kind of attacks as we periodically change the blinking pattern. Given our current decoder design, an attacker only has 1/108(0.00925) chance of correctly predicting the blinking code. For really security critical tags, we could forego interactive latency constraints and ask users to verify multiple successful decodes.
2. Adversary with a BLE device and an LED: The attacker has access to a BLE device and an LED. They could use the BLE device to sniff or emulate the BLE message and use the LED to emulate the blinking pattern. However, if the BLE communication is encrypted, the attacker does not have access to the appropriate blinking code and cannot act as a man-in-the-middle (MITM). A client can potentially behave as a MITM as it knows which

code the tag is about to blink, but this attack can be prevented by requiring clients to authenticate with a server.

3. Unrestricted physical access to the tag (Evil Custodian attack): This refers to a scenario where an attacker could have unrestricted physical access to the system in the past. Even in this scenario, we ensure that the private key of tag is not compromised by storing it in a secure storage. Even if the location of a tag is moved, its identity is still guaranteed. At worst, an attacker could reprogram the firmware, but it would not be able to pass the authentication check at server and could not spoof the tag.
4. Denial of Service (DoS) Attack : A DoS attack is possible by obstructing the line of sight to the LED or physically destroying the LED. These attacks are easily detectable by the user and also don't result in any leakage of sensitive information.
5. Compromised Tag: If the private key of one tag is compromised, an attacker could replicate its behavior. Deploying each tag with a unique private key protects against breaking the security guarantees for all the other tags.

All these security measures fall short in the case where the server is compromised, as an attacker can easily verify its own fake tags. We can design for a secure system only as long as the server is not compromised.

5.2 GLITTER's Protocol

Our objective is not to make the system impossible to spoof, but to make potential adversarial attempts apparent to users. We take lessons from the above analysis to design the hybrid BLE + visual light GLITTER protocol.

Each tag is deployed with its unique key and maintains a large local counter for the number of blinks. During deployment, activation time of the tag is registered at the server. The tag switches from connectable to non-connectable advertisements when a device attempts to pair with it and begins the blinking process. This guarantees that the tag only blinks on demand, and only one device in the vicinity needs to trigger it.

The blink code is generated randomly and any devices in the vicinity will receive an encrypted version of the code over BLE. We use the AES-GCM Message Authentication Code scheme [16] for encryption, as it allows us to generate crypto-text of any small size (to fit our small payload size), unrestricted by block sizes, as well as verify the authenticity of the message and the sender. AES-GCM also fits well with the broadcast mode of communication, since the keys of the GLITTER tag only have to be shared with the server during the initial setup phase, and no intermediate device requires access to them during regular use. One challenge in using AES-GCM is that both the tag and server need to maintain a common counter, for which we could use the blink counter. Communicating this counter value unencrypted gets around the requirement of time synchronization while maintaining our security guarantees. The BLE advertisement contains 8-bits of status flags, the 32-bit plaintext blink counter, a 40-byte encrypted payload and a 128-bit MAC tag that all fit within the payload of a single advertisement.

In its current form, GLITTER can be used by the device to authenticate the anchor. However, it doesn't provide any guarantee that the device can actually see the anchor. An authenticated device (via server) which is in BLE range could trigger an action without actually seeing the LED, so we can only provide a guarantee that a client is in BLE proximity to the tag. Although this could be solved by changing our communication protocol to be session based, it would sacrifice scalability and the low-power properties of the system.

6 Localization and Pose Estimation

Aligning virtual content with the physical world requires that a display device be able to estimate its position in six degrees of freedom (DOF) with respect to the environment, three for location and three for orientation. Not all applications require estimating all six DOFs. For example, to display a flat augmented UI, as shown in Figure 1a, it is sufficient to estimate only two DOFs (obtained from pixel coordinates). For a complete pose estimation as would be required in Figure 1b, many AR systems use gravity as a common reference, which reduces the problem to estimating four DOFs: three for position and one for alignment of the horizontal axes.

6.1 Using an Anchor on a Plane

If we know that an anchor is placed on a horizontal or vertical surface, we can use this extra information to solve for more degrees of freedom. Pixel coordinates of a tag detection give us a ray pointing outwards from the device camera. The anchor could lie at any distance along the ray, and we must accurately estimate this distance to show 3D AR content.

Many AR frameworks have plane detection capabilities, where they use visual features to identify a surface. If we have prior information about the location of the plane and the anchor placed on it, we can find the intersection of the outward pointing ray and the plane surface (called hit testing). Thus, we can estimate the 3D relative position of the anchor, which is sufficient for many AR visualizations, as can be seen in Figure 10. Anchors on a horizontal surface can provide distance, while anchors on a vertical surface can provide complete distance and pose estimation.

6.2 Using Multiple Anchors

We can also estimate the global pose of a camera using $n >= 2$ GLITTER tags at known locations. We start with the pose estimate provided by the AR framework, which tracks the device in a fixed world frame, denoted here as F_{ar}. Our goal is to find the position of the camera in F_{global}, the global coordinate space in which we've specified the GLITTER tag locations. Since most AR frameworks track the camera relative to its position and orientation at startup, F_{arkit} has its origin at some unknown location in F_{global}. However, since F_{arkit} and F_{global} are both gravity-aligned, we only have to estimate the rotation around the gravity axis (the y-axis). This transformation between F_{global} and F_{arkit} consists of a 3-dimensional translation by $[t_x,t_y,t_z]^T$ and a 1-dimensional rotation by θ about the y-axis. It takes the form:

$$M_{trans} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & t_x \\ 0 & 1 & 0 & t_y \\ -\sin\theta & 0 & \cos\theta & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

The AR framework provides the following matrices:

- C: the 3x4 extrinsic matrix that converts points from F_{arkit} to F_{camera} (the camera-rigid coordinate frame)
- K: the 3x3 intrinsic matrix that projects points in F_{camera} into pixel coordinates.

Combining the above, the full equation to project points from F_{global} $[x,y,z]^T$ into pixel coordinates $[u,v]^T$ is:

$$[u',v',w]^T = K*C*M_{trans}*[x,y,z,1]^T$$

$$\therefore [u,v]^T = P_{t_x,t_y,t_z,\theta}(x,y,z) = [u'/w, v'/w]^T$$

P is parameterized by the unknowns t_x, t_y, t_z, and θ.

For each GLITTER tag i we detect, we obtain a data sample consisting of:

- $[x_i,y_i,z_i]^T$: the known location of the tag in F_{global}
- $[u_i,v_i]^T$: the detected pixel coordinates of the tag on the screen

Our goal is to compute values of the parameters t_x, t_y, t_z, and θ to minimize the reprojection error:

$$\underset{t_x,t_y,t_z,\theta}{\text{minimize}} \sum_i \left\| P_{t_x,t_y,t_z,\theta}(x_i,y_i,z_i) - [u_i,v_i]^T \right\|_2^2$$

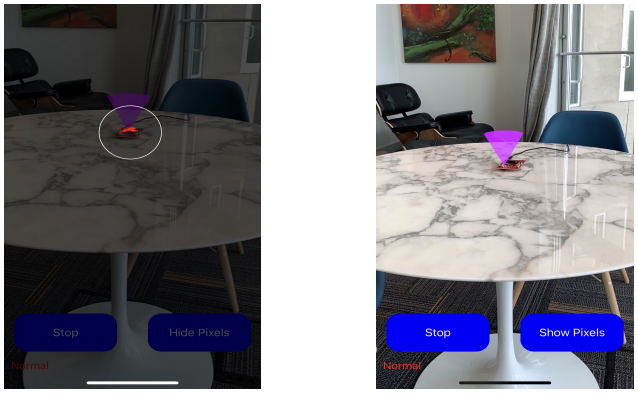
$$\Leftrightarrow \underset{t_x,t_y,t_z,\theta}{\text{minimize}} \sum_i \left\| A_i \cdot [t_x,t_y,t_z,cos\theta,sin\theta]^T - b_i \right\|_2^2$$

$$\Leftrightarrow \underset{t_x,t_y,t_z,\theta}{\text{minimize}} \left\| A \cdot [t_x,t_y,t_z,cos\theta,sin\theta]^T - b \right\|_2^2$$

where A contains the expanded terms of the matrix P. We can replace the trigonometric constants $(\cos\theta, \sin\theta)$ with (a,b) and a constraint $a^2+b^2=1$:

With this formulation, GLITTER tags can be deployed in any constellation (not restricted to a plane) and can even be in different fields of view. This means that a camera can sweep across an area seeing multiple beacons in sequence and still compute camera pose.

7 Implementation



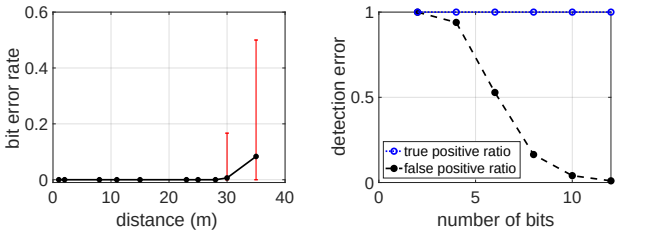
(a) Detected pixels for debugging **(b) Overlaying AR object**

Figure 10: Screenshots from iOS application

GLITTER is implemented in two separate parts: (1) the detector running on an iOS device (iPhone XS and Ipad Pro); (2) the active optical markers which are implemented on BLE-equipped microcontrollers.

7.1 iOS Framework

ARKit captures frames in 1920x1440 resolution in the YCrCb format. We use only the Y channel (Gray scale). We leverage Apple's Metal performance shader on the GPU to achieve real-time performance for both blurring as well as matched filtering. To process each camera frame, our implementation takes 10.238 *ms* on an iPhone XS and 6.815 *ms* on an iPad Pro. All AR object rendering and tracking is handled by ARKit, so the location of the active optical markers is stored as an ARNode object to aid placing virtual objects around it.



(a) BER vs distance **(b) Effect of number of bits on false and true positives**

Figure 11: Signal characterization as a function of distance and number of bits for Indoor/Office environment

7.2 Anchor Implementation

As shown in Figure 4, we build two reference hardware platforms (1) Texas Instruments (TI) LaunchPads for the CC2640R2 and (2) an Adafruit feather board with a Nordic nRF52832.

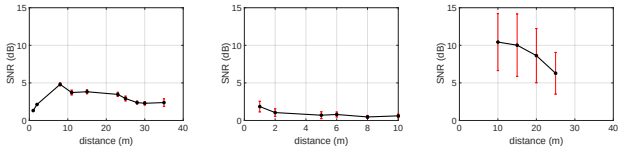
We use a bright 450mcd LED to increase the range from which the active optical markers can be detected. A 5mm diffuser is placed in front of the LED in order to increase the surface area of the LED and boost the contrast by making the Off state dark. On the larger GLITTER box designed for arenas and theaters, we use white LED panels driven by a current driver. The panel is placed behind a neutral density gel film to increase contrast. Our tag application is implemented with TI RTOS or FreeRTOS along with the BLE stacks provided by the manufacturers. Both BLE stacks allow the tag to operate in multiple roles simultaneously, which makes implementing connectible and non-connectible advertisement roles significantly easier.

7.3 Unity and XR Viewer

We integrated the GLITTER iOS framework into two popular platforms for AR development: the Unity engine and Mozilla's XR viewer for AR web applications. The framework operates as a background thread processing images captured by AR Kit while also servicing Bluetooth events. A REST service is used to provide beacon geometry information given Bluetooth UUIDs that can be used by the location solver. For Unity projects, application developers can insert hooks into automatically generated iOS applications that will correct the world origin value anytime a light anchor is correctly decoded. Since the framework continuously processes frames in the background, camera pose information is transparently updated independent of the application's rendering and processing of virtual content. In the case of XR Viewer, we provide a modified version of the browser that has an option to enable GLITTER in the preference panel. Once enabled, the system will trigger and process GLITTER tags in the background to align the origin of any objects rendered from WebXR websites. For websites configured for a particular GLITTER constellation, objects will be transformed into their correct world coordinates. This provides the basis for building extremely simple, persistent multi-user AR scenes that people can load with a URL.

8 Evaluation

In this section, we evaluate microbenchmarks for signal-to-noise ratio and bit error rates that guide our design process followed by end-to-end evaluation of the entire GLITTER system.



(a) Indoor/Office (b) Outside (c) Out of Focus Dark

Figure 12: SNR vs. distance for varying lighting conditions

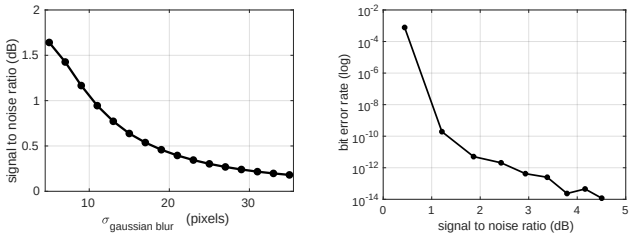
8.1 Experimental Setup

For data collection, we logged data from an active optical marker programmed to generate random test patterns. Our smartphone decoder was modified to record raw frames. The smartphone was either mounted on a tripod for the still cases or held by hand for the handheld cases. The active optical marker would transmit a fixed 6-bit preamble followed by a random 12-bit sequence. At every test point (distance, lighting, etc.), we capture 90 packets each. We transfer the data to a computer and process it using a MATLAB implementation of the demodulator.

8.2 Signal Characteristics

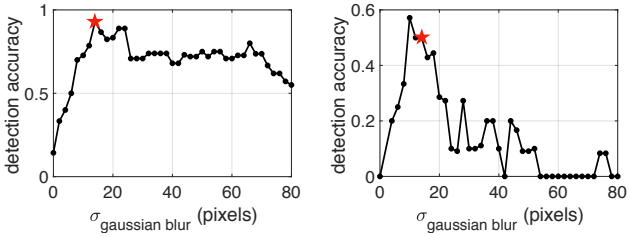
A GLITTER message must have sufficient symbols to be reliably detected, but must also reject false positives caused by noise sources in the environment. Longer messages consume more power, incur more latency and are also more susceptible to errors caused by motion. We varied the number of bits and recorded the true positive and false positive ratios. We can observe from Figure 11b that having fewer bits leads to a large number of false positives. The number of false positives decreases rapidly with an increase in the number of bits. We select an operating point of ten symbols (or bits), as it results in few false positives and is also reasonable in terms of power.

Signal to noise ratio (SNR) is a key parameter for any communication channel. The SNR is a measure of the sensitivity performance of a receiver. To understand how distance affects SNR, we set up an anchor at a fixed location and varied the phone's distance. We then repeat the same experiment across varying lighting conditions. We see in Figure 12 that the peak SNR depends heavily on the environment. This dependence exists because background noise is a function of the environment. In a dark environment, background noise is low and we get high SNR, and the reverse happens in brightly lit environments like the outdoors. Another interesting observation from Figure 12a is that in an office environment, we see an increase in SNR as we go from 1 to 8 meters. This happens



(a) SNR vs blur at a distance of 1m (b) BER vs SNR for indoor/office environment

Figure 13: Plots showing the effect of blur on SNR and relationship between BER and SNR



(a) 1m (b) 5m

Figure 14: Detection accuracy vs Blur for varying distances in the presence of camera shake. ★ denotes the chosen blur level for our system

primarily due to the camera finding less contrast at small distances, which causes higher noise level. Adding darker filters in front of the LED helps reduce this effect at the cost of range. We also see the expected drop in SNR with distance in the office environment, but we get reasonable SNR even at larger distances of 35m where BLE begins to fail. In an outdoor environment with direct sunlight, the drop in SNR with distance is much more severe and falls below acceptable levels from 8m onwards.

Another crucial characterization of our system is the Bit Error Rate (BER) given a particular SNR. Given SNR measurements, we would like to estimate our expected BER to gauge how confident we are about a signal. For measuring BER, we look for an exact preamble match (preamble added for testing) in our data sequence and use it to decode the corresponding data. As one can expect, an increase in distance leads to an increase in bit errors in the decoded signal as the SNR drops. We observe similar behavior as illustrated in Figure 11a for the indoor/office environment. We can decode the data almost perfectly up to a distance of 28m; we start getting bit errors from 30m onwards. As we go further, the size of an anchor (the number of pixels occupied) decreases and at a range of 35m corresponds to an anchor occupying only 5-10 pixels in the image. This shows that GLITTER can decode the LED even when it occupies very few pixels. After 35m we run out of BLE range to conduct further experiments.

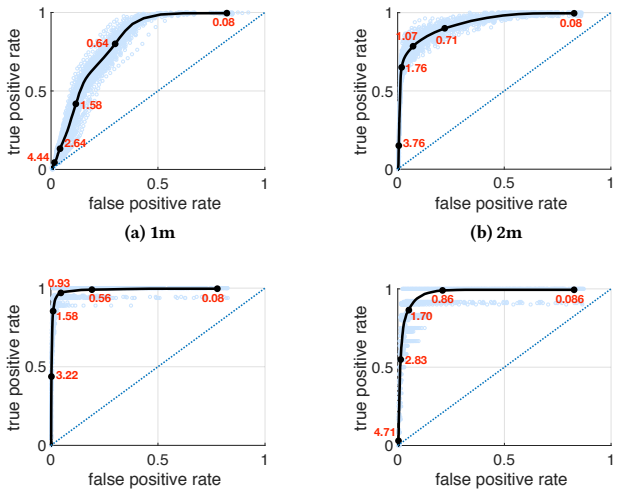
8.3 Channel Characteristics

Figure 13b shows the relationship between BER and SNR. One interesting observation is that our system doesn't follow an Additive White Gaussian Noise (AWGN) model where you would expect a sudden drop in BER as SNR decreases. Instead, we see a more gradual decline, which indicates that standard processing techniques that assume AWGN like matched filters may not perform as well. This insight helped motivate our early bit digitization step in the demodulator processing pipeline.

8.4 Motion

Next, we evaluate the effect of blur on SNR by varying the σ of the Gaussian function. Increasing the blur level leads to a reduction in SNR as the pixels are spread out across the image. The relationship of decrease in received power is shown in Figure 13a.

To evaluate the effectiveness of blurring in our detection pipeline, we tested our system in the presence of hand motion at two different distances of 1 and 5 meters. Ideally, we would like to capture the blur level that maximizes detection rate while minimizing impact on SNR. As we can see from the observations in Figure 14,



(a) 1m (b) 2m (c) 8m (d) 11m

Figure 15: ROC curves for varying SNR threshold at different distances. Values in red denotes SNR thresholds.

with a low level of blur we see very low detection accuracy due to the pixel shift caused by hand motion. Increasing blur leads to improved detection, but only up to a point, at which time the SNR decreases. We see acceptable detection accuracy at around blur levels of 10–12, but in practice can operate at a blur of around 15 to be conservative in terms of false positives.

8.5 Detector Performance

The final output stage from our demodulator is an x and y coordinate on the 2D frame indicating the position of each LED. We use Receiver Operation Characteristic (ROC) curves to explore the detection performance given various SNR thresholds and at different distances. In order to generate these curves we have to classify detected pixels into the following categories:

- True Positive: system detected anchor pixels in a region of the image where there was an anchor.
- False Positive: system detected anchor pixels in a region where there was no anchor present.
- True Negative: system did not detect anchor pixels in an area where there was no anchor.
- False Negative: system detected no anchor pixels in an area where there was an anchor.

The regions in each image were hand labeled with an approximate hand-drawn radius around the LED and used to define the detection zone. We use these metrics to evaluate sensitivity (true positive rate), which denotes the detector's ability to correctly detect the anchor, and specificity (false positive rate), which is a measure of the detector's ability to correctly reject non-anchors.

Each ROC curve is plotted with the true positive rate [1]against the false positive rate [2] while varying an SNR detection threshold. Figure 15 shows multiple curves at different distances. As we previously saw from the BER vs. SNR graph (Figure 13b), having a high SNR leads to lower BER. The ROC curve now captures the deeper implications of these choices in terms of false positives and negatives. The best performance would be achieved by picking the point that is closest to the upper-left corner. Missing true positives results in not detecting a tag, while allowing too many false positives would decrease localization accuracy once the system clusters and finds the LED centroid. Based on these curves, choosing an SNR threshold of 0.85 gives us a decent operating point where we get reasonable detection accuracy without many false positives. We also see that this threshold appears to generalize well across multiple distances.

[1]True Positive Rate = $\frac{\text{True Positive}}{\text{True Positive+False Negative}}$

[2]False Positive Rate = $\frac{\text{False Positive}}{\text{False Positive+True Negative}}$

8.6 Localization Performance

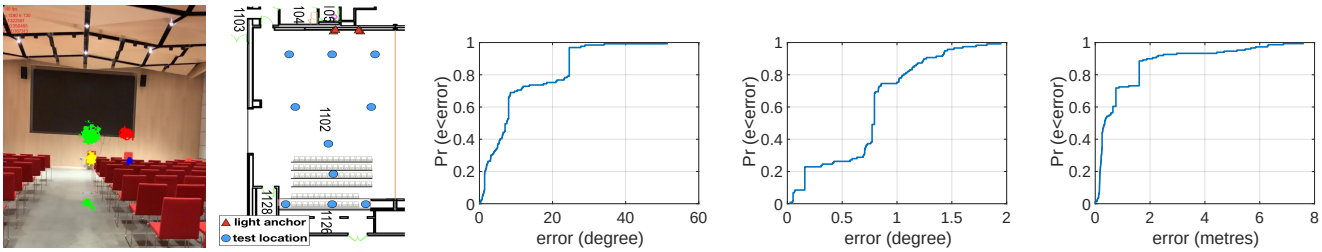
We deployed four GLITTER tags at known locations in a $2m$ x $2m$ square configuration placed at the front of a large auditorium (Figure 16a) and positioned the phone at various locations shown in in Figure 16b. The distance between the tags and the test locations varied from 3 to 25 meters. Figure 16 shows the absolute error in spherical coordinate system between the ground truth and predicted coordinates. Our system incurs low error in elevation and distance estimation, and azimuth estimation suffers primarily due to multipath reflections from the floor. It is worth noting that localization performance is highly dependent on Geometric Dilution of Precision (GDOP). These plots show single estimates that in practice we filter using tracking provided by most AR frameworks.

We specifically chose a challenging environment to highlight the limitations of light based systems, wherein it suffers from multipath due to reflections off of floor, ceiling and walls. Multi-path is specially challenging as none of the traditional signal processing techniques such as SNR based or time of flight based works for this system. This could potentially be solved by performing RANSAC on detected points and only keeping the points which aligns nicely with the known geometry, but this is left as future work for now.

9 Limitations

In its present form, GLITTER cannot guarantee protection against physical man-in-the-middle attacks. For example, an attacker with a photo diode receptor and a BLE repeater could replicate the tag in some other location. At a high level, crude implementations of these attacks are fairly easy to detect by visual inspection which at least increases the barrier to entry for attackers. Even though an attacker can mimic the tag at other locations, our approach provides protection against the introduction of malicious tags. To truly prevent man-in-the-middle attacks, one might consider adding TOF (Time of Flight) capabilities like those found in UWB (Ultra Wide Band) systems, which would complement GLITTER's architecture. Using VLC to identify the location, registering AR content and then authenticating timing with UWB would be a natural extension for highly secure scenarios. UWB alone isn't accurate enough to satisfy the AR content registration functionality of GLITTER.

GLITTER requires both a BLE channel and an LED, which is substantially more expensive than a passive printed tag. For situations in which low-power operation is not a concern, the BLE channel may be removed, allowing the LED to blink codes perpetually. Demodulating GLITTER's optical data requires significant computational overhead. While feasible on modern mobile phones, this approach requires optimizations to run in real-time on earlier generation phones or more constrained embedded targets. GLITTER requires a longer processing latency (400ms) compared with standard optical tags which can be detected in a single frame, increasing sensitivity to camera shake. We believe with improved tracking, it may be possible to decode data without using heavy blurring.



(a) Auditorium (b) Floor Schematic (c) Azimuth Error (d) Elevation Error (e) Distance Error

Figure 16: Localization accuracy CDF wherein we vary the distance between the tags and the phone from 3 m to 24 m. We compare ground truth coordinates with predicted coordinates in a spherical coordinate system.

10 Conclusions and Future Work

This paper introduces GLITTER, an active optical marker system for identifying and precisely tracking multiple tags simultaneously using standard cameras. GLITTER supports up to 108 unique identifiers with a detection latency of 400ms (at 60FPS) and a range of over 10 meters when a phone is held in hand (over 30 meters when the camera is stable). Active optical markers provide an ideal mechanism for attaching AR interfaces and virtual content to objects and physical locations. Leveraging the plane detection algorithms provided by AR toolkits, we are able to use a hit testing to localize tags sitting on prominent surfaces in 3D after a detection. We also show that with two or more anchors at a known location, we can accurately localize the complete pose of the viewing device. An iOS implementation of GLITTER that works with ARKit is available as open source software, enabling third parties to use and extend this work. It can be downloaded at https://github.com/conix-center/LightAnchorFramework. We have already integrated GLITTER into Unity and XR Viewer, two common AR application platforms for mobile devices.

In the future, we believe that we can use approaches like structure from motion and SLAM to perform robust 3D localization and mapping of areas instrumented with active optical markers. If active optical markers are equipped with on-board ambient light sensing, it may also be possible to avoid camera saturation from multiple color channels to improve data rate and reduce detection latency. We eventually envision a miniature energy harvesting peel-and-stick active optical marker that could support perpetual operation.

11 Acknowledgement

This research was generously supported with funds from the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, by the NIST Public Safety Communications Research division and NSF award TWC-1564009.

We are grateful to Chris Harrison and his lab for prototyping of this effort. We would like to thank Bryan Parno for his valuable guidance on our security protocol. Finally, we would like to thank the CONIX PIs, including Mani Srivastava, Rajesh Gupta, Jeff Bilmes and Brandon Lucia for our early discussions at the Karl Strauss Brewing Company (Sorrento Mesa).

References

[1] K. Ahuja, S. Pareddy, et al. Lightanchors: Appropriating point lights for spatially-anchored augmented reality interfaces. In *UIST*. ACM, 2019.
[2] Apple. Arkit. https://developer.apple.com/arkit/, 2019.
[3] D. Balfanz, D. K. Smetters, et al. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*. Citeseer, 2002.
[4] Bluetooth SIG. *Core Specification v5.1*, 1 2019.
[5] C.-H. Chu, D.-N. Yang, et al. Image stabilization for 2d barcode in handheld devices. In *ACMMM*. ACM, 2007.
[6] A. Colin, E. Ruppel, et al. A reconfigurable energy storage architecture for energy-harvesting devices. In *ACM SIGPLAN Notices*, vol. 53. ACM, 2018.
[7] C. Danakis, M. Afgani, et al. Using a cmos camera sensor for visible light communication. In *2012 IEEE Globecom Workshops*. IEEE, 2012.
[8] DecaWave. Decawave. https://www.decawave.com, 2019.
[9] M. Fiala. Artag: Fiducial marker system using digital techniques. In *CVPR*. 2005.
[10] C. Gomez, J. Oller, et al. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9), 2012.
[11] Google. Arcore. https://developers.google.com/ar/, 2019.
[12] J. Grubor, K. Langer, et al. High-speed wireless indoor communication via visible light. *ITG fachbericht*, 198, 2007.
[13] J. Grubor, S. C. J. Lee, et al. Wireless high-speed data transmission with phosphorescent white-light leds. In *ECOC*. VDE, 2007.
[14] HTC. Htc vive. https://www.vive.com/us/, 2019.
[15] IEEE. *SG VLC Project Draft 5C*, 9 2018.
[16] E. Käsper and P. Schwabe. Faster and timing-attack resistant aes-gcm. In *CHES*. Springer, 2009.
[17] A. Kharraz, E. Kirda, et al. Optical delusions: A study of malicious qr codes in the wild. In *DSN*. IEEE, 2014.
[18] D. Knuth. Efficient balanced codes. *IEEE Transactions on Information Theory*, 1986.
[19] T. Komine and M. Nakagawa. Fundamental analysis for visible-light communication system using led lights. *IEEE transactions on Consumer Electronics*, 50(1), 2004.
[20] Y.-S. Kuo, P. Pannuto, et al. Luxapose: Indoor positioning with mobile phones and visible light. In *MOBICOM*. ACM, 2014.
[21] M. Leap. Magic leap. https://www.magicleap.com/, 2019.
[22] F. Lemic, J. Martin, et al. Localization as a feature of mmwave communication. In *IWCMC*. IEEE, 2016.
[23] Y. Ma, N. Selby, et al. Minding the billions: Ultra-wideband localization for deployed rfid tags. In *MobiCom*. ACM, New York, NY, USA, 2017.
[24] J. M. McCune, A. Perrig, et al. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *IEEE S&P*. IEEE, 2005.
[25] Microsoft. Hololens. https://www.microsoft.com/en-us/hololens/, 2019.
[26] E. Olson. Apriltag: A robust and flexible visual fiducial system. In *ICRA*. IEEE, 2011.
[27] B. W. Parkinson, P. Enge, et al. *Global positioning system: Theory and applications, Volume II*. AIAA, 1996.
[28] N. Rajagopal, P. Lazik, et al. Visual light landmarks for mobile devices. In *IPSN*. IEEE, 2014.
[29] R. D. Roberts. Undersampled frequency shift on-off keying (ufsook) for camera communications (camcom). In *WOCC*. IEEE, 2013.
[30] T. Schöps, J. Engel, et al. Semi-dense visual odometry for ar on a smartphone. In *ISMAR*. IEEE, 2014.
[31] Y. Tanaka, T. Komine, et al. Indoor visible communication utilizing plural white leds as lighting. In *IEEE PIMRC*, vol. 2. IEEE, 2001.
[32] S. Thrun, W. Burgard, et al. *Probabilistic robotics*. MIT press, 2005.
[33] Vicon. Vicon. https://www.vicon.com/, 2019.
[34] T. Vidas, E. Owusu, et al. Qrishing: The susceptibility of smartphone users to qr code phishing attacks. In *FC*. Springer, 2013.
[35] M. Vision. Meta2. https://www.metavision.com/, 2019.
[36] D. Wagner, G. Reitmayr, et al. Pose tracking from natural features on mobile phones. In *ISMAR*. IEEE Computer Society, 2008.
[37] G. Woo, A. Lippman, et al. Vrcodes: Unobtrusive and active visual codes for interaction by exploiting rolling shutter. In *ISMAR*. IEEE, 2012.
[38] C. Xiao and Z. Lifeng. Implementation of mobile augmented reality based on vuforia and rawajali. In *ICSESS*. IEEE, 2014.