Card-stunt as a Service: Empowering a Massively Packed Crowd for Instant Collective Expressiveness

Chungkuk Yoo[1*], Inseok Hwang[2†], Seungwoo Kang[3], Myung-Chul Kim[4], Seonghoon Kim[1], Daeyoung Won[1], Yu Gu[4], Junehwa Song[1]
[1]KAIST, [2]IBM Research, [3]KOREATECH, [4]IBM
[1]{ckyoo, shkim, daeyoung, junesong}@nclab.kaist.ac.kr,
[2,4]{ihwang, mckima, yugu}@us.ibm.com, [3]swkang@koreatech.ac.kr

ABSTRACT

Imagine a densely packed crowd that gathers to convey a common message, such as people in a candlelight vigil or a protest. We envision an innovation through mobile computing technologies to empower such a crowd by enabling them simply to hold their phones up and create a massive collective visualization on top of them. We propose Card-stunt as a Service (CaaS). CaaS is a service enabling a densely packed crowd to instantly visualize symbols using their mobile devices and a server-side service. The key challenge toward realizing an instant collective visualization is how to achieve instant, infrastructure-free, decimeter-level localization of individuals in a massively packed crowd, while maintaining low latency. CaaS addresses the challenges by mobile visible-light angle-of-arrival (AoA) sensing and scalable constrained optimization. It reconstructs relative locations of all individuals and dispatches individualized timed pixels to each one so that they can do their part in the overall visualization. We evaluate CaaS with extensive experiments under diverse reality settings as well as under synthetic workloads scaling up to tens of thousands of people. We deploy CaaS to 49 participants so that they successfully perform a collective visualization cheering up MobiSys.

CCS Concepts

•Human-centered computing → Ubiquitous and mobile computing systems and tools; Collaborative content creation; Computer supported cooperative work; •Information systems → Spatial-temporal systems; Mobile information processing systems;

Keywords

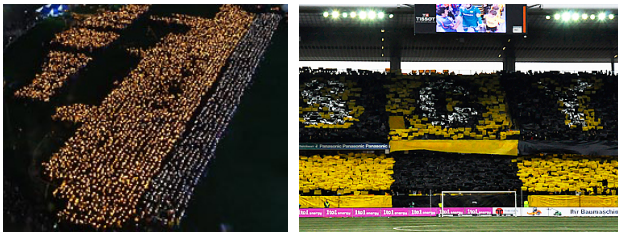
collective visualization; relative localization; visible light communication; optimization; card stunt; mobile-crowd service

*This work was done in part when the first author was on an internship at IBM Research - Austin.
†The corresponding author.

MobiSys'17, June 19-23, 2017, Niagara Falls, NY, USA

ISBN 978-1-4503-4928-4/17/06…$15.00

DOI: http://dx.doi.org/10.1145/3081333.3081357



(a) Candlelight vigil. (b) Card stunt at a stadium.

Figure 1: Massive collective visualization examples.

1. INTRODUCTION

Consider a densely packed crowd gathered in a public space to express a common voice to the community. Such a scene may resemble that seen in Figure 1a, in which over 4000 people participate in a candlelight vigil for memorial and social awareness purposes [7]. Imagine they hold not candles but their phones; each phone automatically glows on and off in a coordinated way such that the entire set of phones collectively visualize a massive symbol or image on top of the crowd.

Why would we want it? It will provide a new prominent medium empowering the crowd's message and attracting the community to listen; people will be able to express their message in a more impactful and visually compelling way by synthesizing their scattered voices, banners, and flags into a single massive one. Fundamentally, it would be an innovation in the way that mobile computing technologies facilitate a new form of instant collaborative crowd activities in the real world, through which people can stimulate their community. In application-wide, such a mobile technology would serve not only public activism but also diverse collective visualization events, such as commercial promotions, flash mobs, sport fans, and collective artistic works, etc.

In this paper, we propose Card-stunt as a Service (CaaS), metaphorically named after the traditional massive sport-supporting activities that could be seen in a stadium (see Figure 1b). CaaS is a new genre of mobile-crowd service to help densely packed crowds express their messages with impact. It enables a crowd to instantly and collectively visualize textual or graphical symbols using their mobile devices. For instance, Figure 2 demonstrates a small group of people using CaaS to perform small-scale, collective visualizations. Imagine a visualization of this kind created by a much larger number of people and at a higher people-per-symbol density.

How would we realize a dynamic, high-quality collective visualization today without CaaS? We refer to the process of planning a traditional card stunt; it requires month-long planning for each venue, rehearsals for specific participants, and often significant budget to hire a professional production [5]. Such a huge cost

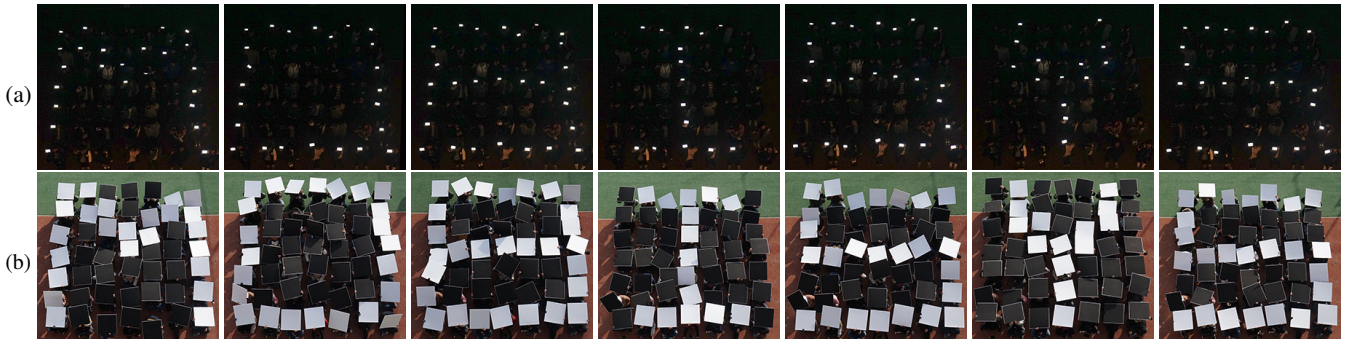


Figure 2: Small-scale demonstrations of CaaS with $7 \times 7 = 49$ participants. (a) Night mode with glowing phone screens. (b) Daylight mode with pieces of cardboard raised or flipped upon individual cues given to each user's phone.

of time, money, and labor would be impractical to most people in our collective visualization scenarios.

CaaS requires none; as soon as a crowd packs into a place like a public square, CaaS lets them instantly create collective visualizations. CaaS automatically determines each user's relative location within the crowd, divides a symbol into a 2-D matrix of pixels, and assigns each pixel to the mobile device at the right location. Each device locally displays the assigned pixel in full-screen. To visualize time-varying symbols, each mobile device may be assigned a timed sequence of pixels. On occasions when sunlight is so strong that a phone's display is indiscernible at full-brightness, CaaS may glow a phone's flashlight instead at the visualization phase. Alternatively, users may improvise physical cardboard with each side colored differently. CaaS can send cues to users' phones, directing them to flip the cardboard and ensure a visual signal even in bright sunlight as shown in Figure 2b.

Our unique challenge to realize an instant collective visualization is how to achieve (1) infrastructure-free, (2) decimeter-level localization of individuals, (3) in a densely packed crowd, while (4) maintaining a low latency for people possibly scaling up to tens of thousands. We do not assume any localization infrastructure like beacons [45] or WiFi access points [41, 49], as those are unlikely in typical gathering places like public squares or wide boulevards. Likewise, existing peer-to-peer localization schemes [21, 58, 68, 72] are not an option, as the one-by-one pair-wise sensing is impractical to handle the high density of people in a crowd.

CaaS tackles those challenges through elaborate mobile-side visible-light angle-of-arrival (AoA) sensing and server-side constrained optimization. To ensure CaaS is suitable for real-world deployment, we carefully devise and implement: (1) fast and robust multi-target visual AoA observations under diverse lighting conditions, partial occlusions, and varying phone orientations when handheld, and (2) scalable and accurate computation strategies to reconstruct the optimal device locations given all the AoA observations. We built a working prototype and conducted extensive experiments to evaluate the step-by-step performances of CaaS under diverse conditions discussed above. We deployed CaaS to 49 co-located participants, where CaaS demonstrated successful reconstruction of individual locations at a typical error radius of 15 cm.

Our contributions are three-fold. First, we set forth the agenda that mobile computing technologies empower people of a common voice and convey a massive real-world impact. Second, we propose CaaS, a mobile service enabling a densely packed crowd to create collective visualizations in an instant, impromptu, and infrastructure-free way. Third, we present our CaaS architecture, novel sensing and computing strategies, and real-world implementation issues to realize accurate reconstruction of densely packed individuals' locations within a reasonably short time.

2. CROWD CHARACTERISTICS

As for the target applications of CaaS to enable large collective visualization, we mainly consider real-world events with densely packed crowds that are relatively stationary and strongly motivated to participate. Examples may be seen in the form of organized fan activities in stadiums [18], commercial promotions [14], or public campaigns [4]. Aside from these existing examples, we also envision that public activism could be a new promising area where CaaS would make an impact. Although large-scale events for public activism are sometimes stereotyped as involving high mobility and unorganized dynamic actions, we note that there exist real protests with peaceful, stationary, and well-organized crowds. Recent examples include the series of candlelight vigils in South Korea in 2016 [15, 20] and the Romanian protests in 2017 [13]. CaaS targets such relatively stationary crowds so that one-time localization results would remain valid for the visualization session.

We analyzed the characteristics of such crowds based on news articles, video footage, journalism literature [9, 13, 15, 20, 36]. We found a few common elements. Notably, their inter-person distances are very small; a participant is most likely surrounded by others within arm's length. Our finding is supported by the studies of political gatherings, (e.g. [36]) which report an average inter-person distance of 0.75 meters. Our analysis also indicates that crowd participants are well-organized and highly motivated. To achieve their goal, they cooperate with a few coordinators in front of them giving actionable cues or chants [9]. Crowd participants will often tolerate significant physical discomfort, as well as financial and temporal cost in order to achieve their goals. For example, the 2016 South Korean candlelight vigils [15, 20] recurred every weekend for five months, during which hundreds of thousands of people gathered, despite freezing temperatures. Similarly, sport fans performing card stunts in a stadium often practice for weeks and pay nontrivial costs (e.g. even a simplest card stunt may cost USD 1 – 3 per person, a higher quality card stunt may cost USD 6 – 29 per person; The rates may further vary depending on the number of symbols and the complexity of the stunts [5]). We believe that CaaS would be highly attractive to such crowds not only in quickly creating massive visualizations but also resulting in very small cost and minimal effort, e.g., holding up their phones overhead for a while, which would be negligible compared to conventional efforts.

2.1 Requirements of CaaS

Imagine a target crowd like those described above. To support an instant collective visualization above such a crowd, CaaS has to: (1) obtain each participant's relative location in the crowd, (2) map each participant to a pixel of the symbol to visualize, (3) let each participant's device know the assigned pixel (or a timed sequence of pixels), and (4) complete (1)–(3) in a short, interactive time.

Figure 3: People raising their phones overhead

In this paper, we address the key technical challenges mainly stemming from (1) and (4). Note that (2) and (3) are straightforward once (1) is complete. For (1) and (4), we derive the following specific localization requirements from our crowd characteristics.

- Infrastructure-free and off-the-shelf phones only. A crowd may gather in a public square or a wide street, which do not generally have fine-grained localization infrastructure like beacons. Crowd members require no special device but their phones.
- High-density crowd support. A collective visualization event deals with densely packed people of hundreds or more, with often an average spacing of only 0.75 meters between each other [36]. Relative localization should support such a density and scale.
- Decimeter-level location accuracy. In such a high-density crowd we should keep individual location errors within one's tiny personal space. Higher errors may cause swapped pixels in the visualization, degrading the readability.
- Fast and scalable reconstruction. Once a crowd forms, the relative localization should be completed in a reasonably short time so that the visualization can start momentarily. Members of both small and large crowds (tens to tens of thousands) should not be required to wait too long to negatively impact interactivity (e.g. 10 seconds unless they are given another task to focus on [53]).

2.2 Limitations of Existing Approaches

Infrastructure-based localization: GPS is commonly available in smartphones. However, its typical location error ranges from 5 to 10 meters [71], which is insufficient to distinguish adjacent people in the crowds. Coin-GPS addresses the indoor unavailability of GPS [54], but the issues of low accuracy and high-gain antenna requirement remain. Higher accuracies are achieved by other techniques relying on pre-installed infrastructure such as WiFi access points [49], beacons [45], and modified LEDs [40, 43, 44]. However, such infrastructure is neither generally available in public spaces, nor is it likely to be installed by crowds on the spot. For example, a state-of-the-art system of visible light sensing [44] achieves 2D localization error of 4 cm, but requires an indoor space with extensive instrumentation, i.e., hundreds of LED lights on the ceiling and hundreds of photodiodes on the floor.

Peer-to-peer relative localization: Peer-to-peer ranging and triangulation among devices is an alternative. Existing techniques have a pair of devices exchange sound chirps or wireless packets and measure TDoA (time difference of arrival) or ToF (time of flight), then convert them into distances [58, 68, 72]. Although those state-of-the-art systems achieve centi- or decimeter level accuracy that is sufficiently accurate for CaaS, the challenges come from a large number of people and limited channel capacity making simultaneous multi-pair sensing difficult. For example, BeepBeep [58] requires all the devices to generate and listen to sound signals one after another. It needs an interval of one second to separate each adjacent sound signal. Chronos [68] takes much less time, i.e., 84 ms, to range a single pair of devices. However, nearby devices around the ranging pair should wait so as not to interfere with this pair's ranging process, as Chronos actively probes all WiFi channels between the pair. In both techniques, the total ranging time for all N devices would be proportional to N; they quickly become intractable in massively packed scenarios like we have assumed.

In short, CaaS deals with high people density where one-by-one pair-wise sensing is infeasible for timely all-pair relative localization, not to mention other issues aggravated in high-density crowds such as elevated channel noise levels and multipath effects.

3. CaaS DESIGN

3.1 Vision-based Collective Localization with Crowd Cooperation

For timely relative localization of all individuals in a dense crowd with no infrastructure and no known reference locations, CaaS adopts visible-light AoA (angle of arrival) observations between cameras and displays of the participants' phones, followed by server-side optimization. This offers key advantages to the CaaS context: multiple pairs can sense simultaneously while hardly interfering with each other. Most smartphone cameras have sufficiently good image resolutions and optical systems capable of high-precision AoA observation from a light source within their FoV (field of view); the multi-path issue is hardly a concern compared to wireless- or sound-based sensing. Collectively processing mobile-side AoA observations, server-side optimization enables all devices' relative locations to be reconstructed in an accurate and timely manner.

3.2 Use Case

Applying the approach of vision-based collective localization, we design CaaS for target crowds described in Section 2 to realize collective visualization under the following use case.

1. Once a crowd packs together in a public square for a collective visualization event, each participant runs the CaaS Mobile Application. Each phone contacts the server and joins the event.
2. The event coordinator's device provides a user interface through which she can choose a symbol to visualize, see how many people have joined, and broadcast a cue to them. The symbols could be pre-determined through the participants' consensus in online communities prior to the event. Once a sufficient number of people have joined, the coordinator triggers a request to each users' phone, instructing them to raise their phones overhead.
3. People raise their phones overhead to maximize the line-of-sight chances among the devices, as shown in Figure 3. The application gives audio-tactile feedback to help a user hold the phone at a preferred orientation.
4. Each phone begins visual observation through its camera, while displaying an encoded identifier on the screen. The application notifies a user to lower the phone when either a certain number of distinct devices have been detected or a timer expires.
5. The observations are sent to the server; the participants wait until the server computes their relative locations. They may keep their arms lowered during server computations.
6. Soon enough (e.g., a few seconds), each phone is assigned a pixel specific to its relative location ensuring that each user provides the correct part of the symbol to be visualized.
7. The phone glows the assigned pixel color on full screen. In bright daylight, users might choose to raise a piece of cardboard upon a cue from the phone. The pixels or cues change synchronously if they are given a timed sequence.
8. The coordinator may choose the next symbols without reperforming the localization as long as the crowd remains stationary.

3.3 Main Technical Challenges

To realize the high-level approach above, we have faced multiple technical challenges in reality. In particular, Section 3.4 through Section 6 discuss our system architecture, key techniques, and experiments to address the following main technical problems:

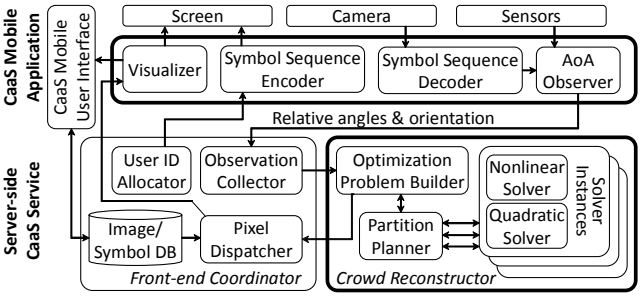


Figure 4: CaaS architecture.

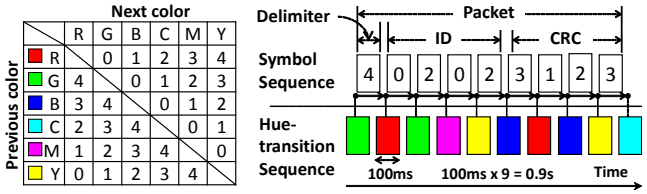
- On the mobile-side, CaaS requires fast and robust multi-target visual observations under diverse lighting conditions, partial occlusions, and handheld phone conditions. Target device screens are seen as too small to observers' cameras, e.g., a 5-inch screen at 6 meters away makes only a few pixels on an observer's FoV. Even worse, we found that extremely diverse lighting conditions in reality make it very hard to identify any content on target screens — too dark under daylight and white-saturated at night. We address these challenges to visual observation in Section 4.
- On the server-side, given all the mobile-side AoA observations, CaaS requires accurate reconstruction of all individual devices' locations quickly enough not to negatively impact interactivity. The computing time for optimal reconstruction should remain short even for tens of thousands of people. Despite accurate mobile-side observations, residual sensor drifts and camera displacements still cause erroneous reconstruction. We also encounter that the reconstruction time rapidly grows for increasing numbers of participants, e.g., more than an hour for a thousand people. We address the server-side challenges in Section 5.

In addition to the above technical issues, there are a number of human and environmental factors that have the potential to strongly impact the success of CaaS. These include (*i*) physical discomfort associated with holding a phone for extended time periods, (*ii*) the dynamicity of crowds, (*iii*) individual height differences, (*iv*) cellular connectivity, and (*v*) time synchronization. In Section 7, we explore the issues in detail and discuss how most of them would be non-critical or transient, based on our real-world observations, our conjecture therefrom, and feasible solutions.

3.4 CaaS Architecture

The CaaS architecture consists of mobile applications and server-side services (Figure 4). Each instance of the CaaS Mobile Application observes local AoAs and displays the assigned pixels. The server-side CaaS service collects the observations, computes all devices' relative locations, and determines the pixel for each device.

CaaS Mobile Application. The *CaaS Mobile User Interface* provides a user with user interfaces to join a collective visualization event, view original images or symbols to be displayed in an event, and give individual cues to raise a phone overhead. It also provides a feature for the coordinators to choose one of the images or symbols that the participants agreed upon beforehand, and to broadcast notifications to the participants such as "Raise your phone." Upon joining an event, the *Symbol Sequence Encoder* retrieves a device-specific identifier (ID) from the server, encodes it into a visual symbol sequence, and displays the code on its screen. At the same time, the *Symbol Sequence Decoder* detects and decodes symbol sequences from the main camera. For each code decoded, the *AoA Observer* measures its AoA from its on-image location and applies sensor compensations. The decoded ID and its AoA values are sent to the server. The *Visualizer* displays the pixels in sequence, which



(a) Symbol encoding scheme. (b) Packet format.

Figure 5: Hue transition-based symbol encoding and packet format.

are assigned to the device by the server. The key challenges are to detect multiple symbol sequences and measure AoAs under diverse lighting conditions, partial occlusions, and varying phone orientations. Section 4 discusses these steps in more detail.

Server-side CaaS Service. The *Observation Collector* collects the mobile-side observations. Once collection is done, the *Crowd Reconstructor* computes individual devices' relative locations using the observations. To achieve this, our *Optimization Problem Builder* builds a constrained optimization problem describing the optimal locations of the devices for the given observation results. The *Partition Planner* intelligently divides the problem into smaller pieces with low inter-dependencies, and invokes one or more *Solver Instances*. Once the optimization is complete, the *Crowd Reconstructor* returns the reconstructed locations to the *Pixel Dispatcher*, which determines a pixel for each device and dispatches it to the corresponding *CaaS Mobile Application*. The key server-side challenges lie within the *Crowd Reconstructor*, which performs sub-meter localizations of densely packed devices within a short and scalable response time. Section 5 presents our basic strategy and advanced issues faced in reality.

4. OBSERVING RELATIVE ANGLES

The key steps in mobile observation are (1) detecting visual symbol sequences seen on an observer's camera, (2) decoding each symbol sequence to identify an observed device, and (3) measuring the AoA from a device. This section presents our symbol coding, real-world issues, AoA measurements, and observing performance.

4.1 Designing Visual Symbol Sequences

Our collective visualization scenarios pose a number of challenges to mobile visual observations. First, the target devices are seen as too small, e.g., a 5-inch screen as far as 9 meters away. Furthermore, this represents a best case, since in reality the target phone is handheld overhead, making its observed representation slanted or partially occluded. Second, the external lighting conditions are highly diverse, ranging from a sunny street to a midnight candlelight vigil. Third, multi-target observations should be completed quickly so that people do not hold their phones overhead for long. These challenges make spatial codes inviable [29, 32, 70]. In our experiments, even a state-of-the-art technique [32] fails to support code detection on a small screen at the envisaged distances, even if occlusions are completely avoided.

Our alternative is to adopt temporally coded full-screen symbols inspired by [52, 61]. We favor a lightweight design in order to enable real-time multi-target detection, and find optimal parameters to ensure a high throughput and thereby quickly complete the observations. We encode data (e.g., device ID) into a sequence of *k*-level quantized hue values. Hue is a component in HSB (Hue Saturation Brightness) color spaces, representing a chromatic element of a color orthogonal to brightness and saturation.

Figure 5b illustrates our packet structure, beginning with a leading delimiter symbol followed by data symbols and CRC (Cyclic Redundancy Check) symbols. Note that we apply differential coding; a symbol is not defined one-on-one for each hue level but for a

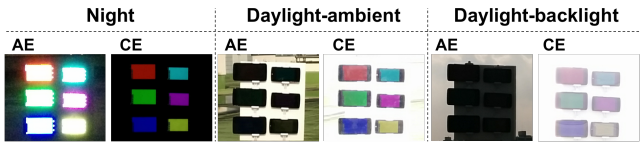


Figure 6: Phone screens under different lighting conditions with (AE: auto, CE: custom-found) exposure settings.

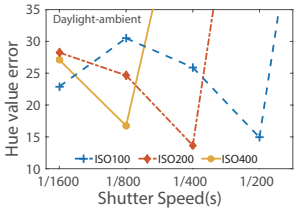


Figure 7: Tuning exposure.

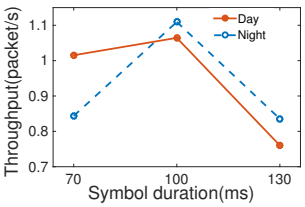


Figure 8: symbol durations.

temporal hue transition. This use of differential coding has the effect of improving throughput. Our early implementations used one-on-one coding, but testing revealed that observing devices can often encounter randomly occurring non-symbol hue sequences that have a valid packet structure, i.e. the sequence begins with the delimiter hue and then contains a sequence of non-delimiter cues; the decoder rejected such false-positive sequence at the CRC checker, but the high frequency of such false-positives requires an extended CRC to ensure that all occurrences are reliably rejected.

By contrast, our revised design with differential coding greatly reduces the occurrences of such false-positive sequences. Since each legitimate packet is expected to change its hue level between every consecutive pair of frames (a scenario that is unlikely to naturally occur), any pixel areas whose hue levels do not change between frames can be rejected early even before reaching the CRC checker. Overall, our differential coding approach has a reduced reliance on CRC when compared to the one-on-one coding approach, achieving an equivalent false-positive rejection rate with much fewer CRC symbols per packet. Thus we can make each packet shorter, resulting in higher throughput. Replacing the single-symbol delimiter with a multi-symbol preamble might be an alternative to early rejection of non-packet sequences, but it makes a packet longer than using differential coding.

Figure 5a illustrates our mapping table for a case of $k = 6$, which maps a hue transition pair onto a base-5 digit, i.e., a symbol. Note that we determined $k = 6$ through extensive experiments, which are discussed in Section 4.2. For the leading delimiter, we assigned the digit "4" among the available five digits. We used the remaining available digits (0, 1, 2, 3) for the device identifier and the CRC. Therefore, the 9-symbol packet with 6-level hue quantization can represent $4^4 = 256$ different device identifiers. The sender repeatedly transmits this packet in full-screen.

4.2 Code Parameters

Exposure settings. In theory, the hue component is defined orthogonally to the brightness component. Still, we found that external lighting conditions often cause the observer to see a distorted hue value that is large enough to result in a wrong quantization level, i.e., a symbol error. Figure 6 shows pictures of phone screens; the one taken under daylight is too dark to recognize the hue, whereas the other taken at night is white-saturated. This is mainly caused by the auto-exposure feature of cameras. When taking a picture, this feature automatically optimizes the ISO value and exposure time based on the ambient brightness. We overcome this issue by enforcing custom exposure settings for three major outdoor lighting conditions—night, daylight-ambient, and daylight-backlight (i.e., camera faces the sun). Figure 7 shows the mean errors of the observed hue values from those originally transmitted. Our experiments show the smallest hue value errors at ISO 200 and 1/400 s exposure time both in night and daylight-ambient conditions, whereas in a daylight-backlight condition, the smallest hue value errors are found at at ISO 100 and 1/400 s exposure time. The errors soar once the exposure is too long to saturate the image sensor. Figure 6 shows the effectiveness of these custom settings.

The CaaS Mobile Application automatically determines the current lighting conditions using the time of day and built-in sensors and manipulates the exposure settings accordingly using Camera2 APIs available in Android 5.0+. Given the phone is outdoors, it is obvious to determine either daylight or night condition from the time of day. To further determine either daylight-ambient or daylight-backlight conditions, CaaS first computes the sun's celestial orientation based on the date, time, and geographic location of a device. Then it sets the daylight-backlight condition if the sun would be seen within the camera's FoV at a current device's orientation. CaaS would be able to adopt an advanced image processing technique to determine whether an image is back-lit [23].

Hue quantization levels. A higher throughput is desirable to help users lower their phones earlier. A finer-grained quantization level (i.e., a higher k) in our coding gives higher bits per symbol. However, a higher k may increase SER (symbol error rate) which degrades the throughput. Thus we need to find a good value of k at which the effective throughput is maximized. Interestingly, we found good k values differ across the display types: OLEDs and LCDs. Experiments in daylight ambient conditions, using a three-meter distance between devices and optimal exposure settings, show the best throughput of 1.11 packets at both $k = 6$ and 7 for OLEDs; $k = 6$ achieves zero SER, whereas $k = 7$ increases the bits-per-symbol but the throughput remains the same, since the increment is nullified by the degraded SER. For LCDs, the optimal k value is 14. Further experiments indicate an optimal k value of 7 (OLED) and 20 (LCD) in both daylight-backlight and night conditions. Since CaaS has no way to know the display type of an arbitrary target display a priori, use of the lower k value for each lighting condition is most appropriate. For our prototype, we select a conservative k value of 6 for use across all lighting conditions

Symbol duration. For given k quantized hue levels, another parameter impacting the throughput is the symbol duration; a shorter symbol duration gives higher bits per symbol but increases SER. Most smartphone displays refresh at 60 Hz [17]. However, most smartphone cameras capture frames at a maximum of 30 fps, where Nyquist-Shannon sampling theorem gives the theoretical minimum symbol duration of 67 ms (= 15 Hz symbol rate). Figure 8 reports the throughput for varying symbol durations. For 70 ms daylight, the higher bits per symbol and SER break even, resulting in comparable throughput with 100 ms. Our final choice is 100 ms due to the real issue of occasional frame drops from cameras, which having 3 frames per symbol is more robust against.

At this symbol duration, a single 9-symbol packet transmission takes 1 second because a the packet is encoded into a sequence of $9 + 1 = 10$ hue levels. However, as discussed earlier, CaaS Mobile Application repetitively transmits the same packets. If 9-symbol packets are concatenated N times, this data stream will be encoded into a sequence of $N \times 9 + 1$ hue levels. Thus, the average time duration per packet will converge to 900 ms as N increases.

4.3 Real-world Issues

Rolling shutter effect. We experienced that observer devices often detect incorrect hue transitions that include a phantom hue level that was not shown either before or after the transition at the target

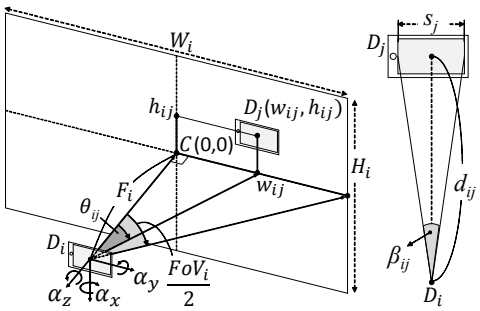


Figure 9: An observer D_i's camera sees a target D_j's display.

screens. This is a result of rolling shutter effects, an anomaly in digital photography whereby a single frame consists of pixels each of which is captured at slightly different times. The reason is that an image sensor usually captures a frame by rapidly scanning the sensing pixels vertically or horizontally, not taking a snapshot of the entire scene at once. Figure 10a shows an example frame affected by rolling shutter. This frame is captured while the shown target device was making an R-to-G transition. Apparently, the reddish pixels around the bottom-left corner are captured slightly before the transition, whereas the greenish pixels around the top-right corner are captured slightly after the transition. The yellowish pixels in the middle are exposed to both colors at the very moment of transition, resulting in averaged colors. Due to the yellowish pixels, an observer device may mis-decode this frame into either R-to-Y or Y-to-G transitions. We filter out this anomaly by exploiting the characteristics of the effect: 1) a wrongly quantized hue level is a mixture of two valid hue levels detected consecutively, and 2) this hue level is seen only from a single frame. Note that we also check the symbol duration to ensure the valid duration of 100 ms (3 frames). But this duration alone is not a reliable measure due to frame drops and inconsistent frame intervals in reality.

Real-time image processing. To ensure the real-time constraint of 33 ms per frame, we implemented the Symbol Sequence Decoder in Android RenderScript for flexible parallelism on multi-core CPUs and GPUs. Our RenderScript implementation processes a single frame in 15.2 ms, whereas our alternative implementation in Android NDK (Native Development Kit) takes 56.5 ms. Note that it could be possible to utilize parallelism by using OpenCL [12] in our NDK implementation, but an OpenCL is unavailable to some phone models, e.g., Google Nexus family, unless rooted.

Our decoder divides a 1280×960 frame into 25600 subframes of 8×6 pixels each. In parallel, each subframe is downsampled by 4 to 1. Taking a raw frame at 640×480 narrowed FoV in some phone models. By contrast, downsampling had a negligible impact on the accuracy. For each subframe, the decoder finds a single quantized hue level dominating this subframe. At each subframe location, a packet is detected upon the transitions of the dominant hue levels that conforms to our symbol definitions in Figure 5, as well as passes CRC. The decoded ID is associated with the subframe location to compute its AoA.

4.4 AoA Measurement

Figure 9 shows the geometry of two devices, the main camera of an observer device D_i and the display of a target device D_j. Our goal for AoA measurement is that D_i measures the horizontal angle θ_{ij} between C (center of D_i's image plane) and w_{ij} (horizontal pixel coordinate of D_j as seen on D_i's image plane). We can derive θ_{ij} as a function of w_{ij} and F_i (focal length of D_i in pixels):

$$\theta_{ij} = \arctan 2\left(w_{ij}, F_i\right), \quad (1)$$

(a) (b) (c)

Figure 10: (a) Rolling shutter effect during a (R to G) transition. (b)-(c) Visual observation experiment setup.

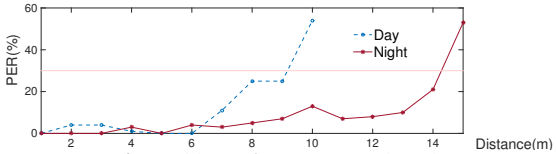


Figure 11: Packet error rate (PER).

where $F_i = \frac{W_i}{2}\left(\tan\frac{FoV_i}{2}\right)^{-1}$, W_i is the pixel width of the image plane, FoV_i is the horizontal FoV in radian available runtime through Android APIs, $\arctan 2(y,x)$ is a two-variable extension of $\arctan\frac{y}{x}$ defined over $(-\pi, \pi]$ range [2].

This is an ideal case when D_i is at a perfect landscape orientation and standing upright, i.e., $\alpha_x = 0, \alpha_y = 0, \alpha_z = 0$. In reality, a device held in a user's hand would have nonzero orientation angles, resulting in wrong w_{ij} and θ_{ij}. To reorient w_{ij} and evaluate the true θ_{ij}, we first define an X-Y-Z intrinsic rotation, where the element rotation angles α_x, α_y, and α_z are obtained by the built-in Android API, the magnetometer, and the gravity sensor. Figure 10c defines the axes of rotation. Each observing device applies Z- and Y- rotation matrices R_z and R_y to the pre-rotation (h_{ij}, w_{ij}, z_{ij}), where z_{ij} is the point where a virtual sphere of the radius F_i intersects with a line originating from (h_{ij}, w_{ij}) orthogonal to D_i's image plane. The post-rotation coordinates are given by:

$$\begin{bmatrix} h'_{ij} & w'_{ij} & z'_{ij} \end{bmatrix}^T = R_x(\alpha_x)R_y(\alpha_y)R_z(\alpha_z)\begin{bmatrix} h_{ij} & w_{ij} & z_{ij} \end{bmatrix},$$

where R_x, R_y, R_z denote the 3-D rotation matrices with respect to x-, y- and z-axis, respectively. Then, we compute the re-oriented θ'_{ij} by (1) and w'_{ij}. The CaaS Mobile Application sends θ'_{ij} to the server-side *Observation Collector*. Note that to make θ'_{ij} compatible with a server-side reference orientation, each device also reports its own α_x along with the observation so that the server normalizes θ'_{ij} with respect to the mean of all devices' α_x.

Vision-based AoA sensing also gives a rough estimate of the physical distance to the observed device. In Figure 9, the distance d_{ij} is given by $d_{ij} = s_j/2\tan\frac{\angle\beta_{ij}}{2}$, where s_j is the screen's physical length and $\angle\beta_{ij}$ is its horizontal angular width seen by D_i. We can obtain s_j from D_j's specification. In reality, D_j may be slanted or partially occluded, resulting in a smaller $\angle\beta_{ij}$ to D_i's view. Thus, d_{ij} is not a definite distance but only an upper-bound that the real distance could be at most. We include d_{ij} as an upper-bound distance constraint for our optimization in Section 5.1.

4.5 Mobile-side Observation Performance

We evaluate the mobile-side observation performance using two Nexus 5s, one for an observer and the other for a target. The display size is 4.95 inches and the camera's FoV is $59.6° \times 46.7°$.

PER in handheld observations. Figure 11 shows the PER (packet error rate) when two devices are hand-held by participants. The observed device transmits 9-symbol packets 100 times for each lighting and distance condition. The PER remains low within a certain

Lighting, orientation	Night Ideal	Day Ideal	Day Random	Day Hand-held
(Avg, σ)	(0.64, 0.44)	(0.53, 0.31)	(1.29, 0.69)	(1.23, 1.03)

Table 1: Observation angle error (degree).

distance, i.e., under 10% until 9 meters at night, under 5% until 6 meters under daylight. As discussed in Section 4.2, the average packet duration is less than 1 second for a 9-symbol packet; even for extended identifier digits supporting 10000+ people, 1.5 seconds will suffice. Thus repeated observations for a few seconds will boost the chances of successful decoding at further distances. Without considering the rolling shutter effect, the PER increases to 33% for observation even at 1 meter away at night.

Accuracy of AoA measurement. We evaluate the accuracy of the observation angle θ_{ij} in controlled settings. We conduct a set of measurements under varying conditions of ground truth angles, distances, ambient lighting, and observer orientations. We measure the error of θ_{ij}, i.e., the absolute difference between an estimated angle and a ground truth angle varying within the camera's FoV.

First, we evaluate the angle errors under the ideal orientation, i.e., two devices heading in the same direction and mounted on landscape tripods (Figure 10b). The ground truth observation angle is set by a protractor at $\pm 25°$, $\pm 20°$, $\pm 10°$, and $0°$ at distances from 1 to 10 meters at every 1 meter increment. This means that we measured the accuracy at 70 different locations under each lighting condition. The average AoA error on 70 different locations is reported in Table 1. We test two lighting conditions: daylight (55000 lux) and night (0 lux). Results indicate that errors in observation angles are typically small, i.e., less than $0.7°$ on average regardless of ground truth angles and lighting conditions, as shown in Table 1.

Next, we evaluate the accuracy at realistic phone orientations under the daylight conditions: (1) an observer device randomly oriented but firmly held on a tripod (Figure 10c), and (2) both devices hand-held by human participants. For (1), we measured the rotation angles after an arbitrary rotation that happened to be: $\alpha_x = -20°$, $\alpha_y = 7.4°$, and $\alpha_z = 7.4°$. For (2), two persons raised their phones overhead and stood at the corresponding observation angle and distance. For each of the two conditions, we measured AoA errors at 70 different locations as mentioned above and obtained the average. Table 1 shows slightly increased errors compared to the ideal cases. These errors may be due to possible compass offset [75]. Section 5.2 addresses server-side compensation for this error.

5. RECONSTRUCTING CROWD LOCATIONS

The CaaS server first collects the mobile-side observations – tuples of (D_i, D_j, θ_{ij}). Each tuple indicates that an observer D_i has visually identified a target D_j at an angle of θ_{ij}. Based on the observations, it then reconstructs the locations of all devices by solving a constrained optimization problem. We present how we formulate the optimization problem and discuss our approaches to improving reconstruction accuracy and scalability for large crowds.

5.1 Formulating an Optimization Problem

We build an optimization problem subject to a set of geometric constraints. The goal is to compute device locations that produce inter-device angles closest to the real observed angles.

Constraints. We begin with a geometric illustration of an observation (D_i, D_j, θ_{ij}), as shown in Figure 12. Device D_i's camera observed another device D_j's ID at a relative angle of θ_{ij}. The coordinates of the two devices (x_i, y_i) and (x_j, y_j) as well as the distance between them are all unknown. In theory, this observation enforces one constraint, which is that D_j should exist at an arbitrary point only on the line originating from (x_i, y_i) towards the direction

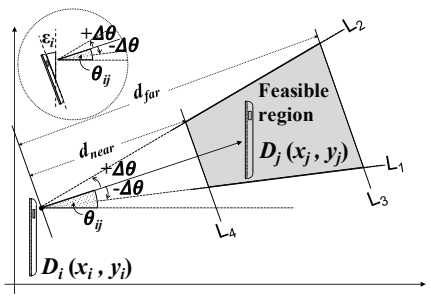


Figure 12: Cartesian plane for constrained optimization.

of D_i's camera at a slope of θ_{ij}. In reality, however, we consider a possible imprecise observation of θ_{ij} and physical limitations. Thus we define a so-called *feasible region* where D_j would exist, which is the trapezoidal area enclosed by four lines (L_1, L_2, L_3, and L_4) highlighted in Figure 12. The rationale for these lines is as follows: (1) L_1 and L_2 are employed to incorporate possible error bounds of θ_{ij}; we set the slopes of L_1 and L_2 to be apart from θ_{ij} by $-\Delta\theta$ and $+\Delta\theta$, respectively. (2) We incorporate the upper-bound distance that comes with each observation by which D_j could be apart from D_i at most as discussed in Section 4.4. Also, D_i and D_j should be apart by at least a single person's torso width. These two distance constraints define L_3 and L_4 as orthogonal lines to $\overline{D_iD_j}$ and distant from D_i by d_{far} and d_{near}, respectively.

Putting them together, we derive the constraints associated with the observation (D_i, D_j, θ_{ij}): four linear inequalities (2) through (5) defining the region enclosed by $L_1 - L_4$:

$$L_1 : \; y_j \geq \tan(\theta_{ij} - \Delta\theta)(x - x_i) + y_i \,, \tag{2}$$

$$L_2 : \; y_j \leq \tan(\theta_{ij} + \Delta\theta)(x - x_i) + y_i \,, \tag{3}$$

$$L_3 : \; x_j \leq \frac{1}{\tan\left(\theta_{ij} + \frac{\pi}{2}\right)} \left\{ y_j - (y_i + d_{fy}) \right\} + (x_i + d_{fx}) \,, \tag{4}$$

$$L_4 : \; x_j \geq \frac{1}{\tan\left(\theta_{ij} + \frac{\pi}{2}\right)} \left\{ y_j - (y_i + d_{ny}) \right\} + (x_i + d_{nx}) \,, \tag{5}$$

where $d_{fx} = d_{far} \cos\theta_{ij}$, $d_{fy} = d_{far} \sin\theta_{ij}$,
$d_{nx} = d_{near} \cos\theta_{ij}$, $d_{ny} = d_{near} \sin\theta_{ij}$.

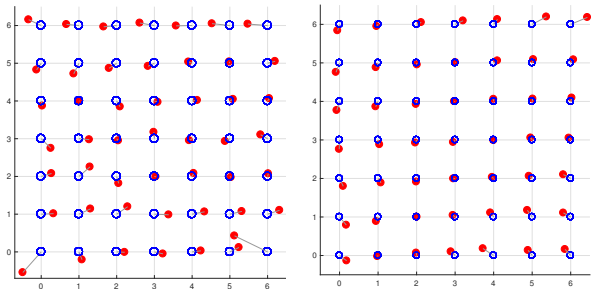
Constraints (4) and (5) place x_j on the left-hand side to deal with a zero slope instead of an infinite slope when $\theta_{ij} = 0$. Having the constraints in linear forms is favorable to lowering the computational complexity. Back to the original problem with N devices, we build the whole constraints set by concatenating all the inequalities for each (D_i, D_j) pair where an observation exists in between.

Cost function. We define a cost function as the gross unsigned difference between the reproduced angles and the observed angles. Consider N devices and the observations among some of them. Let O denote a set of device pairs (i, j) where an observation from D_i to D_j exists. The goal is to find $(x_1, y_1, ..., x_N, y_N)$, minimizing the error between θ_{ij} and the computed slope of $\overline{D_iD_j}$ after reconstruction. The cost function f and its gradient are given by:

$$f(x_1, y_1, ..., x_N, y_N) = \sum_{(i,j)\in O} \left\{ \arctan 2(y_j - y_i,\, x_j - x_i) - \theta_{ij} \right\}^2 , \tag{6}$$

$$\nabla f = \sum_{i=1}^{N} \left(\frac{\partial f}{\partial x_i} \hat{\mathbf{x}}_\mathbf{i} + \frac{\partial f}{\partial y_i} \hat{\mathbf{y}}_\mathbf{i} \right) . \tag{7}$$

Note that f must be differentiable to warrant a gradient. Thus we use a square term for the unsigned angular difference in (6). $\arctan 2(y, x)$ is differentiable by x and y except where $\sqrt{x^2 + y^2} + x = 0$. This singularity is not in our domain because D_i and D_j



(a) Baseline reconstruction results. Mean LE: 0.26 mpd

(b) Camera displacements overlooked. Mean LE: 0.14 mpd

Figure 13: Reconstructed locations of 49 phones placed on 7×7 grids. 1 mpd=75 cm. Blue hollow circles (○) denote ground truth locations. Red solid circles (●) denote the computed locations.

do not overlap, observations are unidirectional, and we define the Cartesian plane to always place D_i to the left side of D_j.

Although we focus on only the "relative" locations, we need one fixed reference point on the Cartesian plane to keep our problem domain bounded. Without loss of generality, we anchor a single arbitrary device at the origin: $x_1 = 0, y_1 = 0$.

Now the reconstruction problem is translated into a nonlinear programming problem to find $(x_1, y_1, ..., x_N, y_N)$, minimizing f in (6) subject to linear inequality constraints (2) – (5) and two linear equality constraints of $x_1 = 0, y_1 = 0$. Several algorithms are available to solve this family of problems [22, 69]. CaaS uses `fmincon`, an off-the-shelf implementation of these algorithms in MATLAB.

5.2 Tackling Error Factors

To represent location accuracy in our collective visualization context, we define a unit *Mean Person Distance* (mpd) — the mean distance between adjacent people on their ground truth locations. We report the individual location error (LE) and the average location error (Mean LE) in units of mpd. Although the physical distance of 1 mpd may vary in different crowds, we refer to an average inter-person distance of 0.75 meters from [36]. We set 1 mpd to be 0.75 m in controlled experiments unless specified otherwise.

Through experiments with real phones, we found that the technique proposed in Section 5.1 results in noticeable localization errors. Figure 13a shows an example reconstruction for 49 phones placed on 7×7 grids. The location errors are shown in units of mpd. Every phone is localized within LE < 1 mpd.

5.2.1 Incorporating Compass Offset

In spite of the mobile-side reorientation of local coordinates, we still observed some errors in the device's horizontal heading α_x due to the gyroscope drift and magnetometer interferences [75]. We found that the local measurements of α_y and α_z are more reliable which the gravity sensor mostly determines.

To compensate possible errors in the devices' α_x readings, we define a variable ε_i that is an observer-specific error between its true horizontal heading and the measured one. For an observer D_i, we presume ε_i is a constant for a short period of time, representing a static offset in its α_x readings. Formally, we revise our cost function f to make it possible to manipulate ε_i within a moderate interval:

$$f(x_1, y_1, ..., x_N, y_N, \varepsilon_1, ..., \varepsilon_N)$$
$$= \sum_{(i,j)\in O} \left\{ \arctan 2 \left\{ y_j - y_i,\ x_j - x_i \right\} - (\theta_{ij} + \varepsilon_i) \right\}^2 \quad (8)$$
$$\text{where } -\Delta\varepsilon \le \varepsilon_i \le +\Delta\varepsilon, \text{ for } 1 \le i \le N \quad (9)$$

The gradient in Equation (7) incorporates ε_i terms, and our linear inequality constraints incorporate the constraint (9). We found $\Delta\varepsilon = 20°$ works for most cases. Figure 13b shows an improved reconstruction by applying ε_i terms.

5.2.2 Incorporating Off-centered Camera

Figure 13b exhibits small but consistent angular shifts by 3–10° in the reconstructed locations. The shifts are slightly counterclockwise from the ground truth. The reason is that the camera lenses are not centered on the phones' body, but displaced by 4.3 cm to 6.5 cm. In landscape view, this creates an effect as if the observer device were slightly aside from the true location.

We resolved this issue by replacing the observer's Y-axis coordinates y_i with $y_i + \delta_i$ in (2) – (5) and (8); δ_i is a per-device constant indicating the displacement of the camera from the phone's center. We approximate δ_i as the half of the phone's screen height given by Android API. Figure 20a shows improved reconstruction (Mean LE: 0.07 mpd) after including camera displacements. Note that the target's coordinates (x_j, y_j) remain the same in the equations because the observer mostly sees the center of the target's display.

5.3 Keeping Computing Time Scalable

Our mobile-side experiments have dealt with a relatively small number of people (or devices) ($N < 50$) due to the practical challenges associated with recruiting a larger number. Still, we can test the Crowd Reconstructor with synthetic workloads of a larger N. We built a mobile-side simulator that (1) generates ground-truth locations for given N simulated people and (2) synthesizes realistic observations among the devices. The ground-truth locations are generated to roughly form a given spatial formation, and have the inter-person distances follow a given statistic distribution, i.e., a normal distribution of $\mu = 0.75$ and $\sigma = 0.2$. The observation syntheses incorporate the real camera's FoV, distance bounds, and imperfect orientations that we learned previously. Using the synthetic observations, we measured the reconstruction time on a server running on moderate hardware – Intel i7 2.8 GHz CPU, 16 GB RAM.

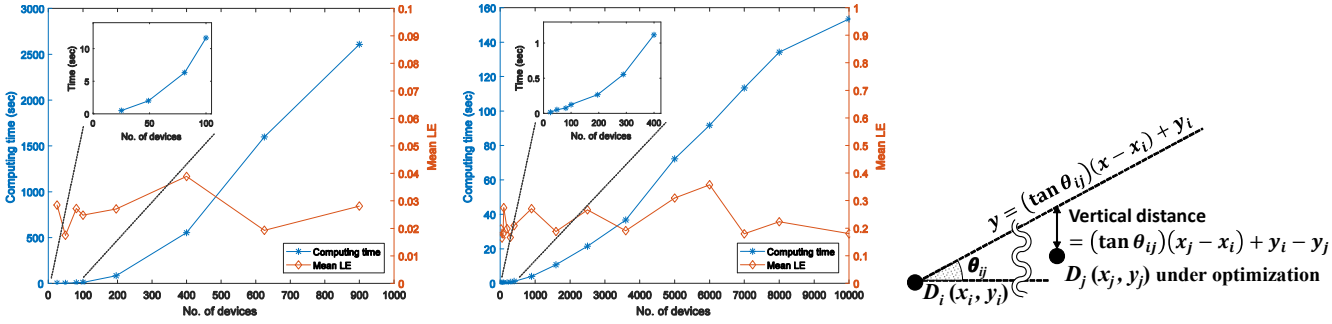
Figure 14a shows that our implementation of Crowd Reconstructor takes significant computing time as N increases, becoming impractical at $N = 200$. This means that, even after mobile-side observations are completed, the people must still wait for many minutes until the reconstruction completes and card stunts are fully ready.

5.3.1 Accelerating with Initial Location Estimates

So far we have bootstrapped the nonlinear optimization algorithm with random initial locations. We observed that initial values more similar to the ground truth locations make the algorithm converge faster. To obtain approximate initial values at much less complexity, we build a linearly constrained quadratic programming problem. Figure 14c illustrates our quadratic cost function, which is the vertical distance between (x_j, y_j) and the line from (x_i, y_i) at a slope $\tan\theta_{ij}$. Our quadratic-formed cost function is given by:

$$f_a(x_1, y_1, ...) = \sum_{(i,j)\in O} \left[\left\{ (x_j - x_i)\tan\theta_{ij} + (y_i + \delta_i) \right\} - y_j \right]^2$$

We used `quadprog`, an off-the-shelf quadratic programming implementation in MATLAB, to find (x_i, y_i)s which minimize f_a, subject to the same constraints as in Section 5.1. Figure 14b shows the computing times of this quadratic programming for $N \le 10000$ as well as Mean LE of the estimated initial locations. These initial locations exhibit much larger Mean LEs by an order of magnitude compared to those achieved by nonlinear optimization (plotted together in Figure 14a). Still, we observe that providing these initial locations to the nonlinear optimization saves an average of 43.5% computing time compared to random bootstrapping.



(a) Nonlinear optimization for $N < 1000$ (b) Quadratic optimization for $N \leq 10000$ (c) Quadratic cost function geometry.

Figure 14: (a), (b): Computing time and Mean LE for varying N. (c): Quadratic cost function.

Figure 15: Visualizations for N=48, 500, 5000 at similar localization errors (0.18<Mean LE<0.21). Red dots and gray dots denote ground truth locations whose pixels are on and off, respectively.

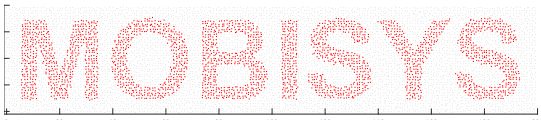


Figure 16: Visualization only with initial locations.

5.3.2 *Diminishing Relative Impacts in Larger Crowds*

So far the reduced computing time is still too long to be practical for N in an order of hundreds or more. We need a different level of breakthrough. Our insight is that, for a larger N, a relaxed reconstruction accuracy may not impact much on the visualization quality. Figure 15 shows the varying visualization quality for different N even at a similar Mean LE. These visualizations are created by CaaS using simulator-generated observations and determining on or off for each device. The resulting pixels are visualized on their ground truth locations. The main factor of varying quality is the pixels-per-symbol. For $N = 48$ only 1 pixel stroke width is allowed per symbol, compared to many pixels in the stroke for $N = 5000$.

In this light, we address that even the initial location estimates would achieve practical accuracies for large Ns. To demonstrate this strategy, we generated the ground truth locations and observations of 8000 people in formation at an aspect ratio of 1:5. Their locations are estimated only by quadratic programming. Then each device is set on or off based on its estimated location. Figure 16 shows the resulting visualization that would appear on top of their ground truth locations. A future question would be: how large N is good enough for this strategy? It may depend on the visualization complexity and the ratio of 1 mpd to the whole crowd area.

5.3.3 *Partitioning into Subproblems for Parallelism*

Section 5.3.2 showed huge time saving by orders of magnitude, e.g., 1000-device reconstruction in less than 5 seconds. Still, the ever-growing computing time is no longer practical for N in the next order of magnitude. For $N = 10000$ it exceeds 2 minutes. It is time to seek parallelism. We exploit our problem's locality from the nature that the mobile-side observations are spatially bounded.

Figure 17 shows such locality for $N = 10000$. The sparsity pattern is a symmetric matrix where (i, j)-th and (j, i)-th elements are set if D_i and D_j has an observation in between. While the original

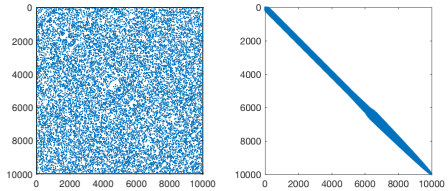


Figure 17: Sparsity pattern of observations for 10000 devices (left) and its RCM reordered matrix (right)

	No. of partitions		1	4	8	16	32
$N = 10000$	Mean LE		0.27	0.26	0.31	0.44	0.46
	Max subprob. time (s)		153	31.7	7.31	1.13	0.45
	% of devices that	$1 \leq LE < 2$	0.10	0.13	1.98	1.69	2.50
		$2 \leq LE < 3$	0	0	0.03	0.03	0
		$LE \geq 3$	0	0	0	0.16	0.10
	No. of partitions		1	16	32	64	128
$N = 40000$	Mean LE		0.26	0.29	0.32	0.50	10.62
	Max subprob. time (s)		1061	21.2	7.52	2.40	0.72
	% of devices that	$1 \leq LE < 2$	0.16	0.23	0.43	8.86	20.6
		$2 \leq LE < 3$	0	0	0.02	0.03	4.04
		$LE \geq 3$	0	0	0	0.03	20.3

Table 2: Trade-offs along with the number of partitions

matrix (left) looks chaotic, applying RCM reordering [26] on the device indices reveals a matrix of strong diagonal locality (right).

We use this reordered diagonal to partition the devices into subgroups, keeping most dependencies therein. This way, only a few observations at the partition borders are discarded. Figure 18a shows how we partition a problem of $N = 10000$ into 4 subproblems of $N = 2500$ each and where the observation losses occur. Figure 18b–18e shows the reconstruction of each subproblem computed independently. While computing the original problem as is took 153 seconds, a subproblem took 31.7 seconds at most. This subproblem computing time further decreases for a greater degree of parallelism, e.g. 7.31 seconds for 8-way. The RCM reordering and putting together the subproblem results take a trivial runtime, e.g., less than 300 ms for 8-way, $N = 10000$.

Finer parallelism creates more partition borders and discards more observations. Table 2 shows a trade-off among the number of partitions, Mean LE, and subproblem computing times. Still most devices are localized at LE < 1 mpd even in small-sized partitions computable in a few seconds, except for 128-way, $N = 40000$.

6. DEPLOYMENT AND EVALUATION

6.1 Reconstruction under Controlled Settings

We evaluate the reconstruction performance under controlled settings using 49 phones of the models: Nexus (5, 6, 5X, 6P) and Galaxy S6, with display sizes of 4.95 to 5.96 inches and horizontal FoVs of 59.6° to 68.1°. To study under different practical condi-

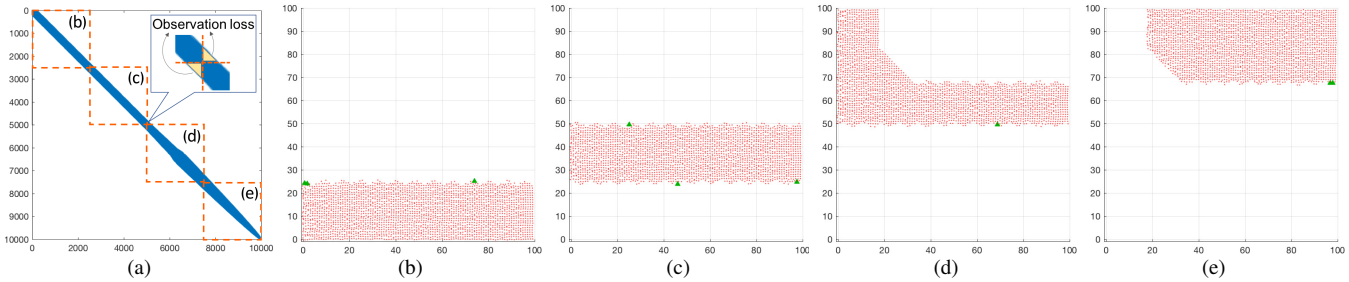


Figure 18: Reconstructing 10000 simulation-generated device locations by 4-way partitioning. (a): Partitions denoted on RCM-reordered sparsity matrix. (b)–(d): Reconstruction results for each partitioned subproblem. Red solid circles (●) denote reconstructed device locations within LE<1 mpd. Green triangles (▲) denote reconstructed device locations where 1 mpd $\leq$ LE $<$ 2 mpd.

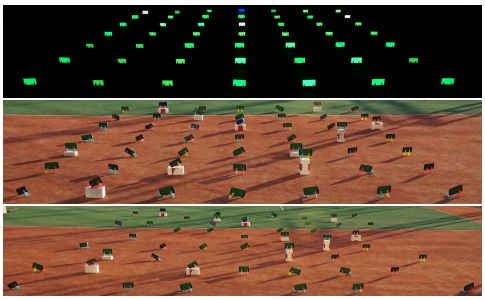


Figure 19: Reconstruction test in varying (orientation, placement, lighting) conditions. Top to bottom: (ideal, uniform, night), (realistic, uniform, day), (realistic, variable, day)

Condition	Mean LE	σ
(I, U, N)	0.068	0.049
(I, U, D)	0.125	0.087
(R, U, D)	0.107	0.067
(R, V, D)	0.105	0.081

(a) Controlled test with 49 phones.

Condition	Mean LE	σ
(R, U, N)	0.186	0.113
(R, U, D)	0.178	0.123
(R, V, N)	0.196	0.133
(R, V, D)	0.168	0.114

(b) Deployment with 49 people

Table 3: Reconstruction errors: Mean LE and σ in mpd. Condition notations: (Orientation {**I**deal | **R**ealistic}, Spacing {**U**niform | **V**ariable}, Lighting {**D**ay | **N**ight})

tions, we conduct experimental trials under different conditions of (i) ambient lighting (day (69000 lux) vs. night), (ii) orientation (ideal vs. realistic), and (iii) device placement (uniformly spaced vs. variably spaced). For the realistic orientations, we applied random rotations around the X or Z axis and height variation, i.e., placing randomly selected devices on boxes 8 – 20 cm high. For device placements, we placed each device on 7×7 grid points with 75-cm spacing in between. For the condition of variable-spaced placements, each device was placed on a randomly selected grid point among 9×9 grid points each spaced by 75 cm. All devices were fixed on the same type of holders. Figure 19 shows the setup. For brevity we use an ordered triple notation (orientation, placement, lighting) to denote an experiment condition. For example (ideal, uniform, night) indicates the conditions of ideal orientation, uniformly spaced placements, and dark lighting at night.

For evaluation, we measure Mean LE in units of mpd as defined in Section 5.2. Figure 20a and 20b show the reconstruction of the 49 phone locations in varying conditions. The reconstruction is close to the ground truth locations; no single device's LE exceeds 0.3 mpd (23 cm). Table 3a shows Mean LEs $\leq$ 0.13 mpd (10 cm) in every condition. The errors are small enough that the the pixel assignments do not deviate from the optimal ones. The lowest error was observed at night because of the large number of observed devices. The average number of observed devices at night is 4.9, while 4.0 and 4.3 devices were observed in (realistic, uniform, day) and (realistic, variable, day), respectively. Counter-intuitively, in (ideal, uniform, day) conditions we observed the fewest, 2.6, because a device would likely occlude those immediately behind it.

6.2 Small Deployment on 49 Participants

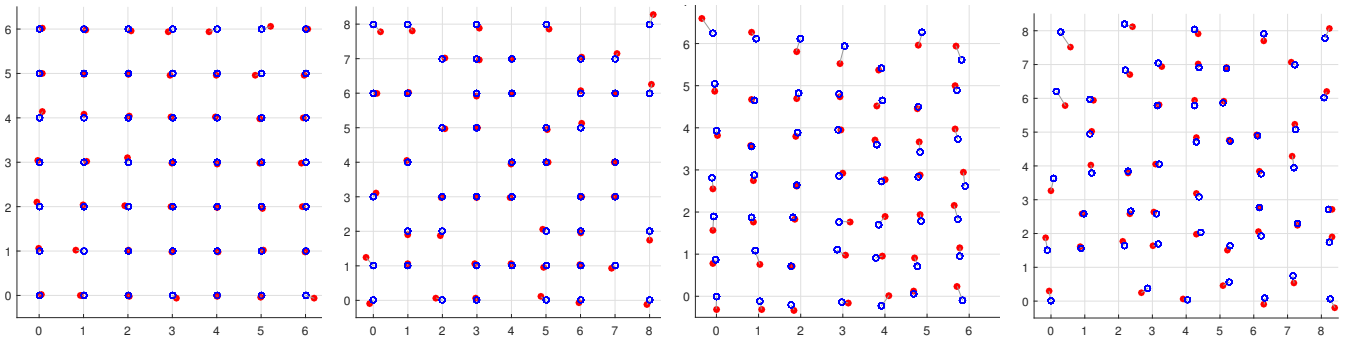
We conducted small-scale deployments using CaaS to reconstruct a crowd of 49 participants and demonstrate CaaS-guided collective visualization. We deployed twice — once under daylight (73200 lux) and another time at night. We recruited the participants from a campus bulletin board. To rule out learning effects, each deployment was conducted with different groups of 49 participants (age: 18 – 35). Everyone voluntarily participated and was compensated with a gift card worth either USD 10 or 15. Those in the daylight group were given the higher valued cards because their visualizations were expected to take more time to flip the cardboard upon the cues given from their CaaS Mobile Application (see Figure 2b). The night deployment took less time because the visualization was automatically done by the glowing phones (see Figure 2a). Our deployment plan was reviewed by the IRB and classified as exempt, meeting the minimal risk research criteria.

The daylight group included 38 males and 11 females (height: 155 – 181 cm). The night group included 32 males and 17 females (height: 155 – 187 cm). For those using iPhones or Android phones below 5.0, we provided our own. The phone models included: Nexus (5, 6, 5X, 6P) and Galaxy (S6, Note5) with display sizes of 4.95 to 5.96 inches and horizontal FoVs of 59.6° to 68.1°.

We demonstrate scenarios in which those 49 participants collectively visualize one letter at a time with their phones (at night) or cardboard (under daylight). Each person was responsible for one of the 49 pixels. We asked the participants to stand on 7×7 grids with 75 cm spacing (see Figure 21a). We also conducted variably spaced cases; each participant stood on a random grid point of their choice out of 81 points of a 9×9 grid (see Figure 21b). Importantly, the ground truth locations of the phones are not aligned on the grids in either case. Each phone is handheld, so that there is a random displacement from the center of the human body. We manually labeled the ground truth locations as seen in the snapshots.

In our deployment, visual observation sessions of each device took from 4.6 up to 30 seconds until 4 to 6 distinct devices were observed. This time included 1) visualizing hue transition sequences and observing AoAs, 2) network delay of sending a cue for "Raise your phone" from the CaaS server to each participant, and 3) human factors from a few sluggish users raising their phones late. We set the observation timeout as 30 seconds to address occasional human factor delays. We believe that the timeout would not be too long for highly motivated people with a specific purpose.

Table 3b lists Mean LEs$\leq$ 0.2 mpd (15 cm) in every condition. No single device's LE exceeded 0.5 mpd (33 cm). Mean LEs slightly grew compared to the controlled cases because (1) the overhead handheld phones underwent minor motions and (2) the ground



(a) (Ideal, Uniform, Night) (b) (Realistic, Variable, Day) (c) (Realistic, Uniform, Night) (d) (Realistic, Variable, Day)

Figure 20: Reconstruction performance. (a)–(b): controlled setting, (c)–(d): real deployments with human participants. Blue hollow circles (○) denote ground truth locations. Red solid circles (●) denote computed locations. Every phone is localized within LE<1 mpd.

(a) Uniform spacing (night) (b) Variable spacing (day)

Figure 21: Two cases of small-scale deployment setting.

truth locations seen from the snapshot picture may have differed from the true locations when each device captured the AoAs. Figures 20c and 20d show the reconstruction of the 49 people's devices and the labeled ground truth locations. Figure 2 show their CaaS-guided card stunts showing "MOBISYS" one letter after another.

7. DISCUSSION

Participants' cooperation: Our use case expects the participants to hold their phones overhead during the initial localization. While this might sound burdensome, we target highly motivated participants as discussed in Section 2. They would feel it insignificant compared to their other efforts. Our experiments showed that the initial localization during which a CaaS user needs to hold the phone overhead takes as low as 4.6 seconds. As for following the coordinator's cues, such motivated participants would be as cooperative as they have been in conventional public gatherings.

Dynamicity of crowd. CaaS is mainly designed for stationary, cooperative, densely packed crowds [9, 13, 15, 18, 20]. As discussed in Section 2, CaaS participants are highly motivated and cooperative to convey their common voices, so they may be willing to remain stationary while using CaaS. Still, some might inevitably move, join or leave while CaaS is in action. An efficient way to handle a newly joined or relocated user is to initiate a localized reconstruction for a subgroup of devices in the vicinity of the new user. Users that have left the visualization could be detected through events triggered when the CaaS Mobile Application is closed on the devices. To enable a newly joined device to detect nearby devices, the CaaS Mobile Applications could sporadically toggle Bluetooth discoverability [46, 57]. Only a tiny fraction of the devices should be discoverable at a time to avoid channel flooding. Detecting a few nearby device identifiers would suffice for the CaaS server to define a spatial subgroup to be reconstructed.

Potential cellular bottleneck. CaaS needs Internet access for each device to send AoA measurements to the CaaS server. Concurrent cellular traffic from tens of thousands of densely packed users would render the nearby base stations irresponsive [39]. Despite the tiny payload (< 1kB) to/from a CaaS Mobile Application, they may produce a short, concurrent burst of messages at the initial localization. We believe that this issue is transient due to the upcoming 5th Generation (5G) Mobile Networks. We discuss the feasibility of Internet access via 5G networks and other alternatives.

- A major goal of 5G networks is to support simultaneous connectivity for a large number of devices. The 5G White Paper from the Next Generation Mobile Networks (NGMN) Alliance [11] specifies the capability of providing simultaneous Internet connectivity to 150000 devices per square km. It also specifies a stadium example with an equivalent density — 30000 simultaneously connected devices in a stadium with 25.6 Mbps average rate per device. Moreover, high-density connectivity load will be further distributed as such an area will most likely be covered by multiple cell towers, each likely owned by different mobile carriers. Even one carrier may operate multiple cell towers in an overlapped manner. We envision that, in the near future with 5G networks operational, packed participants of CaaS in a stadium or in a public square will be able to connect to the CaaS server.
- Adopting ad-hoc network techniques would be an alternative to reduce simultaneous connections to a cell tower. Shafiq et al. proposed a connection-sharing technique to reduce network overload [62]. Notably, they showed its efficiency in a scenario very similar to CaaS (e.g., a stadium). Adopting the connection sharing into CaaS will greatly reduce the simultaneous connections to a nearby cell tower. While a device (namely D_A) is connected to the CaaS server via a cellular network, dozens of nearby devices around D_A send their AoA measurement data to D_A via Bluetooth or WiFi Direct. D_A then sends the collected data as well as its own measurement to the CaaS server via a cellular connection. This way, only a small portion of the devices connect to cellular networks, but all devices still communicate with the server. The individual CaaS messages are very small (<1kB), which are favorable to keep the aggregated message size not burdensome for D_A to send via the cellular connection.
- While CaaS aims at public spaces where rich WiFi infrastructure is unlikely, stadium-specific scenarios may benefit from WiFi infrastructure available at some advanced stadiums. There were 700 WiFi access points in operation at New Orleans Superdome during the Super Bowl in 2013 [27]. IBM Smarter Stadiums [8] and Cisco Connected Stadiums [6] offer dense WiFi connectivity for tens of thousands of fans in a stadium, which will soon be deployed in Atlanta Stadium [3] and Texas A&M Stadium [19].

Time synchronization. To visualize animated symbols, CaaS requires a method to synchronize the local clocks of the participants'

devices. The desired time-sync resolution depends on the dynamicity of card stunt contents. For example, a 30-fps animated image would require a fine-grained time-sync resolution smaller than 33 milliseconds. On the other hand, intermittently changing images such as those in conventional manual card stunts may tolerate even a few seconds of time-sync resolution. Our deployment shown in Figure 2 is the latter case, i.e., showing one character at a time for several seconds and than changing to the next one. In the deployment, we did not observe a noticeable out-of-sync problem even without any time-sync method. Nonetheless, CaaS needs time-sync functionality to support faster animations. Below we discuss a few time-sync options available to off-the-shelf smartphones:

- GPS achieves a typical error in the order of sub-microseconds [48]. This is favorable for CaaS since most scenarios are expected to be outdoor events. An Android device can compute its local clock relative to a global reference clock from the satellites [1].
- Network Time Protocol (NTP) is an alternative for sub-second time-sync resolution, achieving a typical error in the order of tens of milliseconds (WiFi) or hundreds of milliseconds (4G) [47].
- Cellular networks provide a built-in time-sync feature, Network Identity and Time Zone (NITZ) [10], but it is not an option for CaaS because its typical error is in the order of minutes.

Facing similar directions: The initial localization works best when the participants face similar directions. The visible light observation is not omnidirectional but constrained within moderate angular ranges; e.g., Nexus 5's camera has an FoV of 59°, and the display exhibits dimmer brightness at peripheral viewing angles. Having the phones see similar directions creates a much higher probability that one phone's camera and another one's display are in the line of sight. This would not be an issue because CaaS will likely target crowds whose participants are naturally facing similar directions, e.g., the stage, the coordinator, or the arena [9, 18, 20]. For further assistance, the CaaS Mobile Application provides sensor-assisted audio-tactile feedback to hold the phone at a certain 3-D orientation, i.e., the display faces forward and the main camera faces backward. Our experiments showed that each phone sees not only the one immediately behind it but many phones within its FoV and a certain range, achieving simultaneous multi-target observations.

Different heights of participants: Even after the phones are aligned, one may argue that not many screens may be seen in a device's FoV due to the different heights of people. We find this hardly a problem. Given a smartphone camera's typical vertical FoV of 46°, an observer device is able to see other devices behind it as long as the height difference is within $\pm$32 cm for a horizontal distance of 0.75 meters; the tolerable height difference grows for people more than 0.75 meter away. For the height distribution of U.S. adults, a 32-cm interval centered at the median covers 98% of the same-gender adults, and 92% of whole adults [16]. In our deployment, we found no problem from height differences. Stepped floors in a stadium may amplify the height differences, but the audio-tactile feedback may consider the floor slope to tilt properly. The coordinator UI may include a simple pointing feature to estimate the floor slope.

Participant feedback: The CaaS Mobile Application shows the symbol to be visualized, so that a participant can opt out anytime. It may be unusual though, as we assume motivated crowds that agreed to the symbols in online communities prior to the event.

8. RELATED WORK

Stitching small displays. Technologies for stitching multiple displays have been proposed in the HCI and mobile computing literature. However, such techniques are poorly suited to the scenarios addressed by CaaS. Phone as a pixel [61] and Where's my pixel [42] use a global camera with an omniscient view to see all participants' phone screens within its FoV. In CaaS scenarios, such a camera would need to be high enough to have a holistic view of the crowd. Requiring such a mid-air camera would greatly increase the barrier to entry, in direct opposition to our objective to offer a commodity service that does not require the installation of additional hardware into public spaces. MagMobile [33] uses specialized hardware attached to all devices to detect the spatial arrangement between two collocated devices in very close proximity up to a few centimeters. Again, CaaS should support unmodified off-the-shelf smartphones. Other approaches require structured light [56], fiducial markers [60], short-range infrared [51], programmable tiles [59] and so on. MovieTile [55] stitches displays by having users repeat pinching gestures for every adjacent device to determine their relative directions. Such a pair-wise gesture would not be user-friendly, fast, nor precise.

Relative localization of distributed devices. We discussed closely related works for peer-to-peer ranging in Section 2.2. In addition to these previously described works, APIT [31] is an influential early work in wireless sensors using radio connectivity among sensor nodes. A large volume of works are based on the radio connectivity principle [74], dual-mode radio [21], or acoustic signals [30]. However, within large-scale densely packed devices, existing connectivity-based approaches would suffer from excessive noise, multipath interferences, and signal attenuation. Event-driven approaches inject events to the system and utilize the detection patterns to infer the individual locations [63, 64, 73], but they require dedicated event generators/detectors such as a moving aerial vehicle emitting focused light beams. While CaaS scenarios address mostly stationary people, moving object tracking or interpersonal ranging techniques [25, 38, 65, 35, 37, 34] may complement CaaS to promptly address individuals joining and leaving.

Facilitating public activism. In the last decade, online social media and mobile connectivity have influenced public activism [24, 28, 66, 67]. Still it remains unclear if social media really facilitates public activism and supports users in the changing of wider societal perceptions [50]. CaaS may complement such online influences to facilitate public activism by attracting real-world attention.

9. CONCLUSION

This paper presented Card-stunt as a Service (CaaS), a service enabling a densely packed crowd to instantly visualize symbols collectively using their mobile devices and server-side services. CaaS features novel robust mobile visual observation and scalable optimization techniques to achieve high-density, infrastructure-free, and fast crowd reconstruction in real environments. Our implementation is robust and effective, demonstrating decimeter-level accuracies in real 49-person deployments, regardless of devices orientation and placement, and of lighting conditions. Our scalable computing strategy exhibits steadily low computing time even for simulated groups of tens of thousands of devices. Beyond the scenarios mainly discussed, CaaS would open up creative opportunities for people to express their ideas in a massive and prominent way.

10. ACKNOWLEDGEMENTS

We thank our shepherd, Dr. Sarah Clinch and the anonymous reviewers for their valuable comments. The corresponding author is Inseok Hwang. This work was partly supported by the National Research Foundation of Korea(NRF) grants funded by the Korea government(MSIP: Ministry of Science, ICT & Future Planning) (No. 2017R1C1B1010619 and 2017R1A2B3010504).

11. REFERENCES

[1] Android API, getFullBiasNanos. https://developer.android.com/reference/android/location/GnssClock.html#getFullBiasNanos(). Accessed: April 2, 2017.

[2] atan2. http://linux.die.net/man/3/atan2. Accessed: December 8, 2016.

[3] Atlanta stadium with IBM smarter stadiums. https://www-03.ibm.com/press/us/en/pressrelease/46116.wss. Accessed: April 14, 2017.

[4] Autism speaks card stunt. http://www.cardstunts.com/autism-speaks-card-stunt/. Accessed: April 8, 2017.

[5] Card Stunts by Kivett Productions. http://www.cardstunts.com/. Accessed: December 8, 2016.

[6] Cisco connected stadium Wi-Fi solution. http://www.cisco.com/c/dam/en_us/solutions/industries/docs/sports/c78-675063_dSheet.pdf. Accessed: April 14, 2017.

[7] Guinness record-setting candlelight vigil. http://english.hani.co.kr/arti/english_edition/e_national/687413.html. Accessed: December 8, 2016.

[8] IBM smarter stadiums. https://www.ibm.com/sports. Accessed: April 14, 2017.

[9] Iceland fans together with national team performs "Viking war chant". https://www.youtube.com/watch?v=BM6U1tmDjUA. Accessed: April 8, 2017.

[10] Network identity and timezone (nitz); service description. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=576. Accessed: April 8, 2017.

[11] NGMN Alliance. 5g white paper. https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf. Accessed: December 8, 2016.

[12] OpenCL framework. https://www.khronos.org/opencl/. Accessed: April 6, 2017.

[13] Protesters create giant Romanian flag. http://www.wsj.com/video/protesters-create-giant-romanian-flag/6EA2237F-2C08-4422-AD55-4BE0D09B637A.html. Accessed: April 8, 2017.

[14] Schneider's 80th anniversary. http://www.cardstunts.com/schneiders-80th-anniversary/. Accessed: April 8, 2017.

[15] South Korean protesters march against president again. http://edition.cnn.com/2016/11/12/asia/south-korean-protest-president-park/. Accessed: April 8, 2017.

[16] Statistical abstract of the united states: 2011. U.S. Census Bureau.

[17] SurfaceFlinger and Hardware Composer. https://source.android.com/devices/graphics/arch-sf-hwc.html. Accessed: December 5, 2016.

[18] Texas A&M completes world's largest card stunt. http://tamu.247sports.com/Bolt/AM-completes-worlds-largest-card-stunt-40896549. Accessed: April 8, 2017.

[19] Texas A&M's Kyle Field fiber for the future, stadium tech report. http://www.corning.com/media/worldwide/coc/documents/TA&M_SINGLEPAGE_LOW.pdf. Accessed: April 14, 2017.

[20] Thousands call on South Korea's Park to step down. http://edition.cnn.com/2016/11/05/asia/south-korea-president-protests/. Accessed: April 8, 2017.

[21] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner. Virtual compass: relative positioning to sense mobile social interactions. In *Proceedings of the 8th International Conference on Pervasive Computing (Pervasive '10)*, pages 1–21. Springer, 2010.

[22] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[23] C.-L. Chin and C.-T. Lin. Detection and compensation algorithm for backlight images with fuzzy logic and adaptive compensation curve. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(08):1041–1057, 2005.

[24] A. Choudhary, W. Hendrix, K. Lee, D. Palsetia, and W.-K. Liao. Social media evolution of the egyptian revolution. *Communications of the ACM*, 55(5):74–80, 2012.

[25] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury. Did you see bob?: human localization using mobile phones. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom'10)*, pages 149–160. ACM, 2010.

[26] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th national conference*, pages 157–172. ACM, 1969.

[27] J. Erman and K. K. Ramakrishnan. Understanding the super-sized traffic of the super bowl. In *Proceedings of the 2013 Internet Measurement Conference*, pages 353–360. ACM, 2013.

[28] S. González-Bailón, J. Borge-Holthoefer, A. Rivero, and Y. Moreno. The dynamics of protest recruitment through an online network. *Scientific reports*, 1, 2011.

[29] T. Hao, R. Zhou, and G. Xing. Cobra: color barcode streaming for smartphone systems. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12)*, pages 85–98. ACM, 2012.

[30] M. Hazas, C. Kray, H. Gellersen, H. Agbota, G. Kortuem, and A. Krohn. A relative positioning system for co-located mobile devices. In *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys'05)*, pages 177–190. ACM, 2005.

[31] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom'03)*, pages 81–95. ACM, 2003.

[32] F. Hermans, L. McNamara, G. Sörös, C. Rohner, T. Voigt, and E. Ngai. Focus: Robust visual codes for everyone. In *Proceedings of the 14th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys'16)*, pages 319–332. ACM, 2016.

[33] D.-Y. Huang, C.-P. Lin, Y.-P. Hung, T.-W. Chang, N.-H. Yu, M.-L. Tsai, and M. Y. Chen. Magmobile: enhancing social interactions with rapid view-stitching games of mobile devices. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia (MUM'12)*, number 61. ACM, 2012.

[34] I. Hwang, H. Jang, T. Park, A. Choi, C. Hwang, Y. Choi, L. Nachman, and J. Song. Toward delegated observation of kindergarten children's exploratory behaviors in field trips. In *Proceedings of the 13th International Conference on*

Ubiquitous Computing (UbiComp'11), pages 555–556. ACM, 2011.

[35] I. Hwang, H. Jang, T. Park, A. Choi, Y. Lee, C. Hwang, Y. Choi, L. Nachman, and J. Song. Leveraging children's behavioral distribution and singularities in new interactive environments: Study in kindergarten field trips. In *Proceedings of the 10th International Conference on Pervasive Computing (Pervasive'12)*, pages 39–56. Springer, 2012.

[36] H. Jacobs. To count a crowd. *Columbia Journalism Review*, 6(1):37, 1967.

[37] H. Jang, S. P. Choe, I. Hwang, C. Hwang, L. Nachman, and J. Song. Rubberband: augmenting teacher's awareness of spatially isolated children on kindergarten field trips. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp'12)*, pages 236–239. ACM, 2012.

[38] J. Jun, Y. Gu, L. Cheng, B. Lu, J. Sun, T. Zhu, and J. Niu. Social-loc: Improving indoor localization with social sensing. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys'13)*, number 14. ACM, 2013.

[39] J. Jurgensen. Concert crowds flounder in digital dead zones. *The Wall Street Journal*, 2014. Accessed: December 8, 2016.

[40] Y.-S. Kuo, P. Pannuto, K.-J. Hsiao, and P. Dutta. Luxapose: Indoor positioning with mobile phones and visible light. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom'14)*, pages 447–458. ACM, 2014.

[41] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. SchilitShow. Place lab: Device positioning using radio beacons in the wild. In *Proceedings of the Third International Conference on Pervasive Computing (Pervasive'05)*, pages 116–133. Springer, 2005.

[42] C. Lewis, A. Chandler, and J. Finney. Where's my pixel? multi-view reconstruction of smart led displays. In *The Fifth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM'11)*, pages 83–89. IARIA, 2011.

[43] L. Li, P. Hu, C. Peng, G. Shen, and F. Zhao. Epsilon: A visible light based positioning system. In *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI'14)*, pages 331–343, 2014.

[44] T. Li, Q. Liu, and X. Zhou. Practical human sensing in the light. In *Proceedings of the 14th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys'16)*, pages 71–84. ACM, 2016.

[45] K. Liu, X. Liu, and X. Li. Guoguo: Enabling fine-grained indoor localization via smartphone. In *Proceeding of the 11th International Conference on Mobile Systems, Applications, and Services (MobiSys'13)*, pages 235–248. ACM, 2013.

[46] C. Luo and M. C. Chan. Socialweaver: collaborative inference of human conversation networks using smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys'13)*, number 20. ACM, 2013.

[47] S. K. Mani, R. Durairajan, P. Barford, and J. Sommers. Mntp: Enhancing time synchronization for mobile devices. In *Proceedings of the 2016 Internet Measurement Conference*, pages 335–348. ACM, 2016.

[48] J. Mannermaa, K. Kalliomaki, T. Manstén, and S. Turunen. Timing performance of various GPS receivers. In *Frequency and Time Forum, 1999 and the IEEE International Frequency Control Symposium, 1999., Proceedings of the 1999 Joint Meeting of the European*, volume 1, pages 287–290. IEEE, 1999.

[49] A. T. Mariakakis, S. Sen, J. Lee, and K.-H. Kim. Sail: single access point-based indoor localization. In *Proceedings of the 12th International Conference on Mobile Systems, Applications, and Services (MobiSys'14)*, pages 315–328. ACM, 2014.

[50] D. McCafferty. Activism vs. slacktivism. *Communications of the ACM*, 54(12):17–19, 2011.

[51] D. Merrill, J. Kalanithi, and P. Maes. Siftables: towards sensor network user interfaces. In *Proceedings of the 1st international conference on Tangible and Embedded Interaction (TEI'07)*, pages 75–78. ACM, 2007.

[52] K. Miyaoku, S. Higashino, and Y. Tonomura. C-blink: a hue-difference-based light signal marker for large screen interaction via any mobile terminal. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology (UIST'04)*, pages 147–156. ACM, 2004.

[53] J. Nielsen. *Usability engineering*. Elsevier, 1994.

[54] S. Nirjon, J. Liu, G. DeJean, B. Priyantha, Y. Jin, and T. Hart. Coin-GPS: indoor localization from direct GPS receiving. In *Proceedings of the 12th International Conference on Mobile Systems, Applications, and Services (MobiSys'14)*, pages 301–314. ACM, 2014.

[55] T. Ohta and J. Tanaka. Movietile: interactively adjustable free shape multi-display of mobile devices. In *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, number 18. ACM, 2015.

[56] T. Okatani and K. Deguchi. Easy calibration of a multi-projector display system. *International journal of computer vision*, 85(1):1–18, 2009.

[57] D. O. Olguín, B. N. Waber, T. Kim, A. Mohan, K. Ara, and A. Pentland. Sensible organizations: Technology and methodology for automatically measuring organizational behavior. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):43–55, 2009.

[58] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys'07)*, pages 1–14. ACM, 2007.

[59] J. Rekimoto. Squama: a programmable window and wall for future physical architectures. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp'12)*, pages 667–668. ACM, 2012.

[60] A. Schmitz, M. Li, V. Schönefeld, and L. Kobbelt. Ad-hoc multi-displays for mobile interactive applications. In *Eurographics (Areas Papers)*, pages 45–52, 2010.

[61] J. Schwarz, D. Klionsky, C. Harrison, P. Dietz, and A. Wilson. Phone as a pixel: enabling ad-hoc, large-scale displays using mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'12)*, pages 2235–2238. ACM, 2012.

[62] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, S. Venkataraman, and J. Wang. A first look at cellular network performance during crowded events. In *ACM SIGMETRICS Performance Evaluation Review*, volume 41, pages 17–28. ACM, 2013.

[63] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke. A high-accuracy, low-cost localization system for wireless

sensor networks. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SenSys'05)*, pages 13–26. ACM, 2005.

[64] R. Stoleru, P. Vicaire, T. He, and J. A. Stankovic. Stardust: a flexible architecture for passive localization in wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys'06)*, pages 57–70. ACM, 2006.

[65] A. Symington and N. Trigoni. Encounter based sensor tracking. In *Proceedings of the thirteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'12)*, pages 15–24. ACM, 2012.

[66] S. Valenzuela. Unpacking the use of social media for protest behavior the roles of information, opinion expression, and activism. *American Behavioral Scientist*, 57(7):920–942, 2013.

[67] O. Varol, E. Ferrara, C. L. Ogan, F. Menczer, and A. Flammini. Evolution of online user behavior during a social upheaval. In *Proceedings of the 2014 ACM Conference on Web Science (WebSci'14)*, pages 81–90. ACM, 2014.

[68] D. Vasisht, S. Kumar, and D. Katabi. Decimeter-level localization with a single WiFi access point. In *Proceedings of 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI'16)*, pages 165–178, 2016.

[69] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical programming*, 107(3):391–408, 2006.

[70] A. Wang, Z. Li, C. Peng, G. Shen, G. Fang, and B. Zeng. Inframe++: Achieve simultaneous screen-human viewing and hidden screen-camera communication. In *Proceedings of the 13th International Conference on Mobile Systems, Applications, and Services (MobiSys'15)*, pages 181–195. ACM, 2015.

[71] P. A. Zandbergen. Accuracy of iphone locations: A comparison of assisted GPS, WiFi and cellular positioning. *Transactions in GIS*, 13(s1):5–25, 2009.

[72] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda. Swordfight: Enabling a new class of phone-to-phone action games on commodity phones. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12)*, pages 1–14. ACM, 2012.

[73] Z. Zhong and T. He. Msp: multi-sequence positioning of wireless sensor nodes. In *Proceedings of the 5th international Conference on Embedded Networked Sensor Systems (SenSys'07)*, pages 15–28. ACM, 2007.

[74] Z. Zhong and T. He. Achieving range-free localization beyond connectivity. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*, pages 281–294. ACM, 2009.

[75] P. Zhou, M. Li, and G. Shen. Use it free: Instantly knowing your phone attitude. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom'14)*, pages 605–616. ACM, 2014.