Mobile Collaborative Video

Kiarash Amiri*, Shih-Hsien Yang+, Aditi Majumder+, Fadi Kurdahi*, and Magda El Zarki+

*Center for Embedded Computer Systems, University of California, Irvine, CA 92697, USA
+Donald Bren School of ICS, University of California, Irvine, CA 92697, USA

Fig. 1. Left: Texas Instruments DLP Pico Projector. Right: Samsung Galaxy Beam phone with an embedded pico projector.

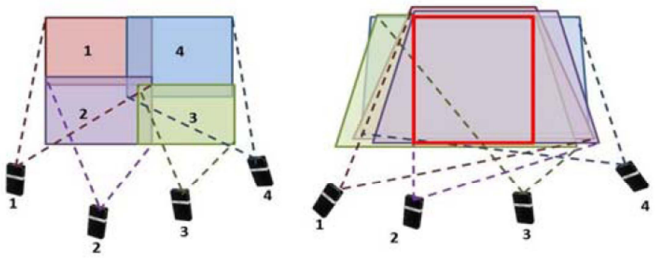

Fig. 2. Improvement of display quality using a federation of projectors. Left: Projections from four mobile devices are tiled with small overlaps to create a display that has almost 4 times the resolution of a single projection. Right: Projections from four mobile devices are superimposed on each other to create a display with 4 times the brightness of a single projection.

***Abstract*—The emergence of pico projectors as part of future mobile devices presents unique opportunities for collaborative settings, such as entertainment, specifically video playback. By aggregating pico projectors from several users it is possible to enhance resolution, brightness, or frame rate. In this paper we present a camera-based methodology for the alignment and synchronization of multiple projectors. The approach does not require any complicated ad-hoc network setup amongst the mobile devices. A prototype system has been setup and used to test the proposed techniques.**

I. Introduction

The pico projectors, which are often less than an inch thick, are the first wave of ultra-portable projectors in the market (Figure 1). It is a response to the emergence of advanced portable devices such as smart phones, portable media players, and high-end digital cameras, which have sufficient processing power and storage capacity to handle large presentation materials but little real estate to accommodate larger display screens. Pico projectors allow projecting larger digital images onto common viewing surfaces, like a wall or table, for extended periods of time. Just recently, we have found pico projectors to be incorporated into mobile phones. For instance Samsung has started marketing the first cell phone with an embedded projector in it (Figure 1). These pico projectors hold a great potential for usage amongst the younger generation. Embedded pico projectors are predicted to be the primary device to be used by younger people for sharing multimedia content and applications [8].

However, what will soon become apparent is that there exists a huge gap between the image/video quality users are expecting from these devices and that offered by pico projectors. Though low-power LED based illumination technology (uses 1.5 watts) and extreme miniaturization (6mm thick and 4-5oz in weight) of the pico projectors have made their presence in mobile devices possible, this has come at the cost of a severely reduced display quality. As opposed to standard presentation projectors with 3000 lumens brightness and at least 2Megapixel resolution, pico-projectors are around 10-15 lumens in brightness (300 times reduction) and around 0.25 Megapixel in resolution (25 times reduction). This indicates that users would probably be unable to view multimedia at the desired brightness and resolution using these projectors.

A. Main Contributions

In this paper we present mobile collaborative video which offers a solution to this problem. Unlike other display devices, projections from a federation of mobile devices can be overlapped or overlaid to alleviate the low display quality. For example, if multiple pico-projectors are superimposed on top of each other, a brighter display results. Or, if multiple pico-projectors are tiled with small overlaps across adjacent projectors, a higher resolution display results (Figure 2). Thus, if we consider a group of users intending to view or share some video content, their mobile devices can be put together to create a federation that can provide a much higher quality display than what is possible by a single device. Such *mobile collaborative video* can thus allow multiple users to view and share media in a much more acceptable fashion than is possible with any alternate mobile display technology. However, achieving such mobile collaborative video poses a few key challenges.

First, the federation of devices needs to be temporally synchronized to display the same frame at the same time. The most natural choice for communication to achieve this synchronization would be the wireless mobile network. Unfortunately, such networks are often fraught with congestion and delays that prohibits their use in time critical operations that needs to be done at a granularity of 30 fps (frames per second) or higher. However, fortunately, most mobile

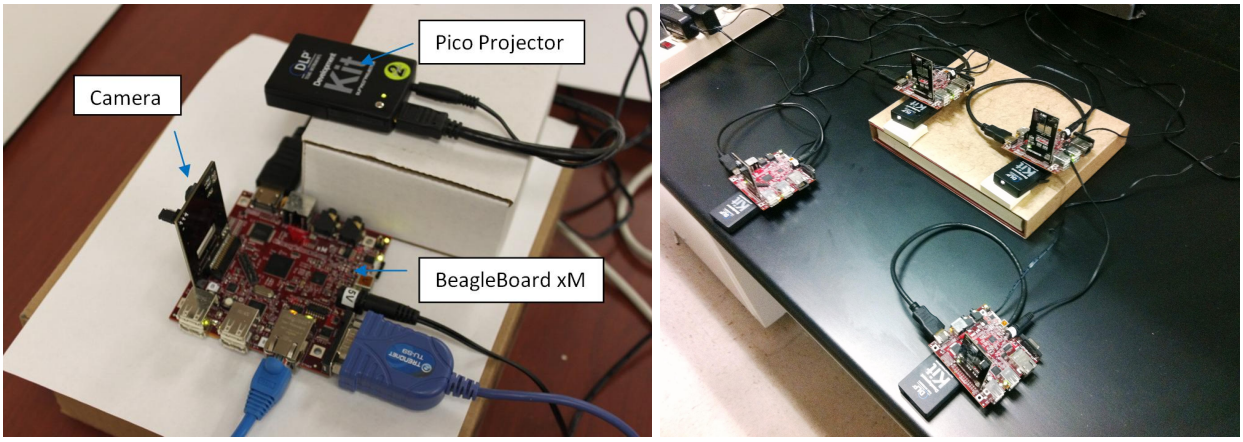

Fig. 3. Left: A single pico-projector setup with a projector, the development board, and the camera. Right: Setup of 4 tiled pico projectors.

devices today come equipped with cameras. We propose the use of such a camera on the device that looks at the projected display and provides visual feedback thus providing an alternate communication channel to achieve the temporal synchronization. We propose novel temporal synchronization schemes for both superimposed and tiled federation of projectors. We also propose ways to detect when the system drifts from synchronization and re-synchronize it back. Finally, we propose a distributed version of the synchronization method which can run as a software application on each separate device to achieve collaborative video.

Second, the images from the multiple projectors must be spatially registered i.e. a point on the display that receives contribution from multiple devices should project the same content and color. Such a spatial registration can be achieved by the same camera looking at the display. For this purpose we import existing registration techniques from the domain of large multi-projector displays to this much smaller scale display formed by our federation of mobile devices. However, these integration is not trivial since they were explored in the context of tiled displays only and not superimposed ones. We propose methods to extend these techniques for superimposed displays as well.

Though mobile collaborative video can reduce resource (e.g. power, bandwidth) consumption per device, when considered in aggregation it comes at the cost of an overhead. Finally, we provide a cost-benefit analysis of this tradeoffs between personal vs aggregated resources. This allows us to identify the optimal configurations (number of projectors and how they are arranged) to pool the resources together in the most beneficial manner. We demonstrate all the above with real prototypes developed in our lab and cross-validate theoretical results with practical implementation.

This work is the first effort to improve the quality of mobile projection based displays using a federation of devices. Though mobile devices today often come with more than one camera, these are usually not positioned to look at the projected display. However, we hope that the benefits of mobile collaborative video would motivate next generation devices to provide such cameras. Since mobile collaborative video can be achieved with very inexpensive cameras, it will barely affect the cost of the mobile devices.

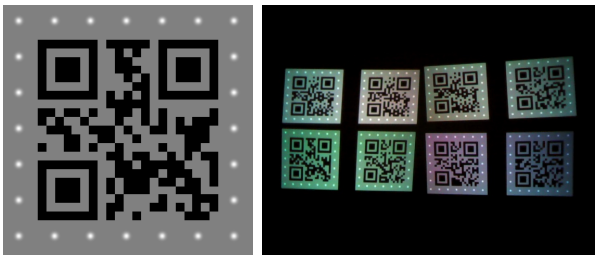

Fig. 4. Left: A QR code augmented with gaussian blobs. Right: QR codes projected in a tiled display.

II. System Overview

We have developed a prototype for a projection-enabled mobile device (will be referred to as device in the rest of the paper) consisting of a Texas Instruments DLP Pico Projector 2.0 Development Kit, a 3-MegaPixel Leopard Imaging Camera board, and a Beagleboard-xM development board. (Figure 3). Note that the camera field of view is usually much larger than the projector field of view. Each development board is connected to a LAN for networking purposes. Then, we create our prototype for mobile collaborative video by putting together four such devices (Figure 3). We use two different configurations – in one the projectors are tiled with small overlaps at the boundaries to increase resolution and in the other they are completely superimposed with each other to increase brightness. Note that for each configuration, initially devices are roughly positioned based on the configuration by the user (Figure 3) and the accurate registration is done by the algorithm.

Assuming we have n devices, our first goal is to register the devices. For this we use a distributed registration scheme used in multi-projector displays [6], [9], [11]. Since we modify the patterns used in this method for our video synchronization, we explain the method in brief first. This method is developed for tiled displays and uses QR (Quick Response) codes [4] augmented with some gaussian blobs as shown in Figure 4. First, the projectors encode information required to achieve registration in the augmented QR codes and display them.

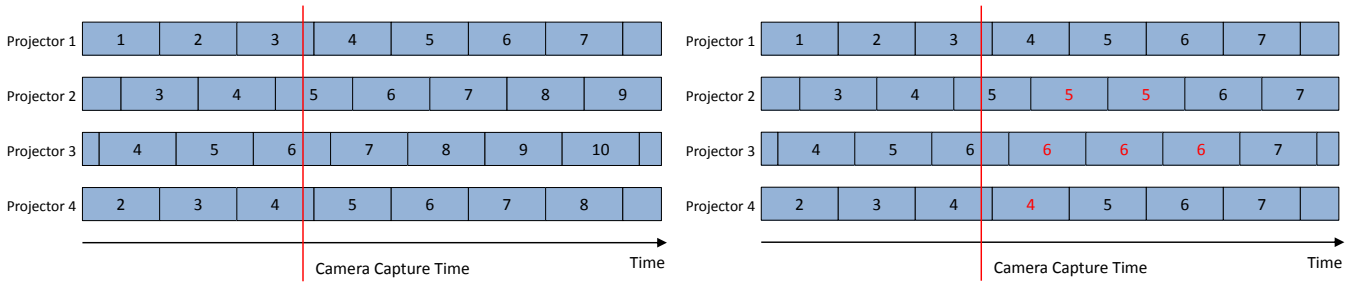

Fig. 5. Left: Frames displayed by 4 out-of-sync projectors. At a given time, the master unit captures an image, and projectors 1,2,3 and 4 are displaying frames 3,5,6 and 4 respectively. Right: After the time difference is propagated to all 4 devices, projectors pause accordingly. All the projectors are synchronized when displaying frame 7. Here we assume zero network latency.

Each device displays four such codes placed in a fashion such that the adjacent device's camera sees at least one of these if it has reasonable overlap with its field-of-view. The cameras on the devices then capture the image of these QR codes to achieve registration using [9]. This returns an ID i, $1 \leq i \leq n$, for each projector and the homography it needs to warp its image so that the display formed by the federation is seamless.

However, the method needs to be modified for overlaid configuration (Section IV). This configuration has not been tried before since in tiled displays using table top projections, brightness improvement by overlaying is not required. So, we first describe our video synchronization method for the tiled configuration. Then, we describe the changes required to adapt both the registration and the synchronization to the superimposed configuration.

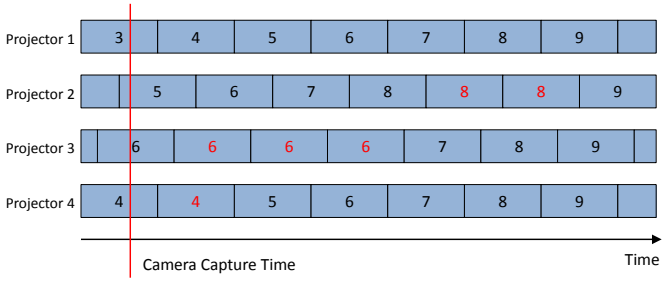

Fig. 6. Effect of network latency on time taken to synchronize all n devices. Here, the message for D_2 arrives 3 frames later than in Figure 5. The device can still synchronize in frame 9 instead of frame 7. Thus, network latency does not prohibit synchronization, but merely delays it.

III. Tiled Configuration

In this section, we describe our camera-based video synchronization method for a federation of pico projectors in a tiled configuration. Our goal here is to synchronize the clocks of the two devices which would allow us to synchronize the video and the audio simultaneously. Further, this would also not need us to play audio from the various devices to get them synchronized. First we consider the simplest scenario where only one device has the camera pointed towards the display, can see the entire tiled display, and captures video at a standard rate of 30fps. Subsequently, we present modifications to the method to arrive at a solution where each device can have the camera seeing only a part of the entire display and can capture at a different frame rate than others. Thus we arrive at a completely distributed SPMD (single program multiple data) algorithm that can be run on each of the devices to achieve the synchronization in a distributed manner.

A. Single-Camera Based Centralized Synchronization

Let us consider a system with n devices. Each device is denoted by D_i, $1 \leq i \leq n$. In this simplest scenario, we assume only one mobile device have a camera looking at the entire projection area to see the images projected by all the devices. This device acts as the master unit and runs the centralized synchronization algorithm calculating the time difference between all the devices. To start the synchronization process, each projector projects a sequence of encoded images. We call this sequence of images as *synchronization sequence*. Each encoded image contains the ID of the device and the current frame number. To encode these information we use the QR codes. The detailed description of the QR encoding is elaborated in Section III-A1.

Each device D_i projects the synchronization sequence. Note that the start time for projecting the synchronization sequence is different for different devices. Figure 5 illustrates an example scenario for 4 out-of-sync projectors. The camera on the master device captures an image that contains the frames projected by all the projectors, indicated by the red line. The captured image is then processed to identify the projector displaying the minimum frame number and hence the maximum frame lag. The goal is to synchronize all the other devices with this device with maximum lag – hence we call this unit as the *reference* unit. Next, the master unit computes the frame difference L_i for each device D_i and communicates it via the network. Each D_i then stalls for next L_i frames to achieve synchronization with other devices. Thus, the synchronization error between any two devices is less than one frame (Figure 5).

Note that the amount of time taken to achieve the synchronization across all the devices is dependent on the time M_i taken by the message from the master to reach the device D_i. However, if we assume the maximum instantaneous network latency is $\max(L_i)$, as shown in Figure 6, a network latency does not prohibit synchronization, but merely delays it. The maximum time that can be spent in achieving synchronization

can be as large as $\max(L_i) + \max(M_i)$.

1) Integrating with Registration Techniques: In the previous section, we explained temporal synchronization of the video frames following which the video plays by displaying parts of the same frames from the different devices at any time. However, note that to make the display seamless, we need to assure that the multiple devices projecting at any pixel in the overlap region should project the same content. Otherwise, ghosting, more formally called geometric misregistration, results. Further, the overlap regions are brighter due to contributions from multiple projectors. These photometric variations should also be compensated. So, we need to apply geometric and photometric registration techniques across the multiple devices to get a seamless collaborative video. Figure 7 and 18 show the images before and after registration.

To achieve registration across the devices, we adopt a technique by Roman et al [9] used in the domain of multi-projector displays. Like us, [9] uses projector-camera pairs and presents a distributed registration technique. As the method starts, each device projects a QR code with information (like its IP address, resolution) embedded in it. The code is augmented by some blobs in the surrounding "quiet zone" which are used to detect correspondences across devices that are critical for achieving the registration (Figure 4). The blobs are embedded in a manner so that the quiet zone is retained after the binarization of the QR code in the decoding phase.

First, The embedded information in the QR codes is used to decipher the total number of devices, identify the configuration of the display (the number of rows and columns) and the location of each device in the array. Second, the embedded blobs are used to find the correspondence and subsequently the homography across adjacent projectors. Finally, a radially cascading method is used to register the images across the multiple projectors geometrically. The homography is also used for edge blending across the overlaps. Thus, note that when projecting the synchronization sequence using the QR codes, the information required for registration does not change. Only the frame ID number changes which is used for achieving the temporal synchronization.

2) Handling Sub-Nyquist Camera Capture Time: We have so far assumed instantaneous capture from the camera which is impossible in reality. When considering non-zero capture times, in order to assure that the projector projects the same content during the time taken for the camera to capture a frame, the camera frame rate needs to be at least 60 fps. However, in most practical systems, the camera and projector frame rate are comparable, usually a video rate of 30fps. Therefore, there is a high chance that the capture duration from the camera spans more than one projector frame. During the synchronization period, this implies that the camera captures more than one QR code with different frame numbers, overlapping with each other, within the same capture time. Hence, the QR code cannot be decoded to get the desired frame number.

We alleviate this problem by alternating the placement of the QR codes in two different non-overlapping spatial regions in consecutive frames. Thus, the QR code is not at the same position in any two consecutive frames. Hence, when the captured image spans across multiple frames, the camera sees two codes from a projector instead of one (Figure 8). Both the codes can now be clearly deciphered. They both contain a frame number, and we only choose the smallest frame number for the delay computation. Note that this does not affect the registration since the information related to registration remains identical among both the QR codes from a projector. In fact, with the increased number of blobs from the two QR codes, we are able to get a larger number of correspondences which are better distributed spatially leading to a higher registration accuracy.

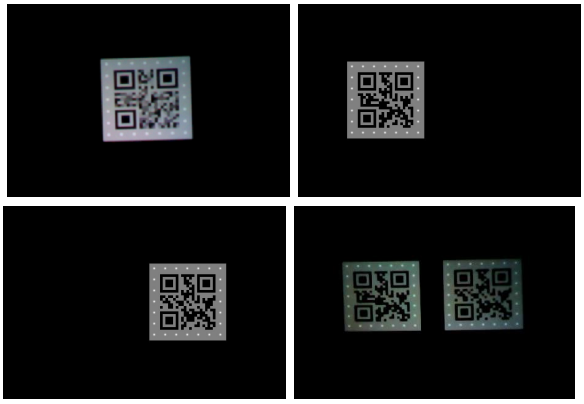

Fig. 8. In scanline order: (a) Overlapping of codes from two different frames when captured by a camera sampling at sub-nyquist rate; (b) the code projected by first frame of the projector; (c) the code projected by the second frame at a different location; (d) the codes captured by the camera when both the frames overlap the sub-nyquist camera's one frame time. Note that the codes are not overlapping now and can be deciphered easily by the master device.

3) Achieving Subframe Synchronization Granularity: When using a standard video camera with capture speed of 30 fps, we can achieve temporal synchronization at a granularity of less than a frame. However, to achieve a granularity of synchronization of a fraction of a frame, we can use high speed cameras where the camera can have much more than Nyquist capture rate of 60fps. High speed cameras today are becoming common and are not that expensive [1], [2]. In this section, we explain how our temporal synchronization method can be modified to use a high speed camera and achieve sub-frame synchronization.

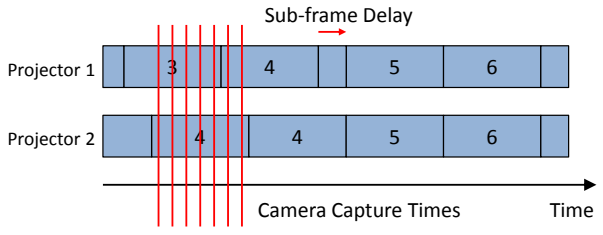

Fig. 9. Example of sub-frame synchronization.

Let us consider the frame rate for the projectors to be f and that of the high speed camera to be m times the display frame rate (Figure 9). Hence, the master unit can capture m frames within one display frame. The master device has to

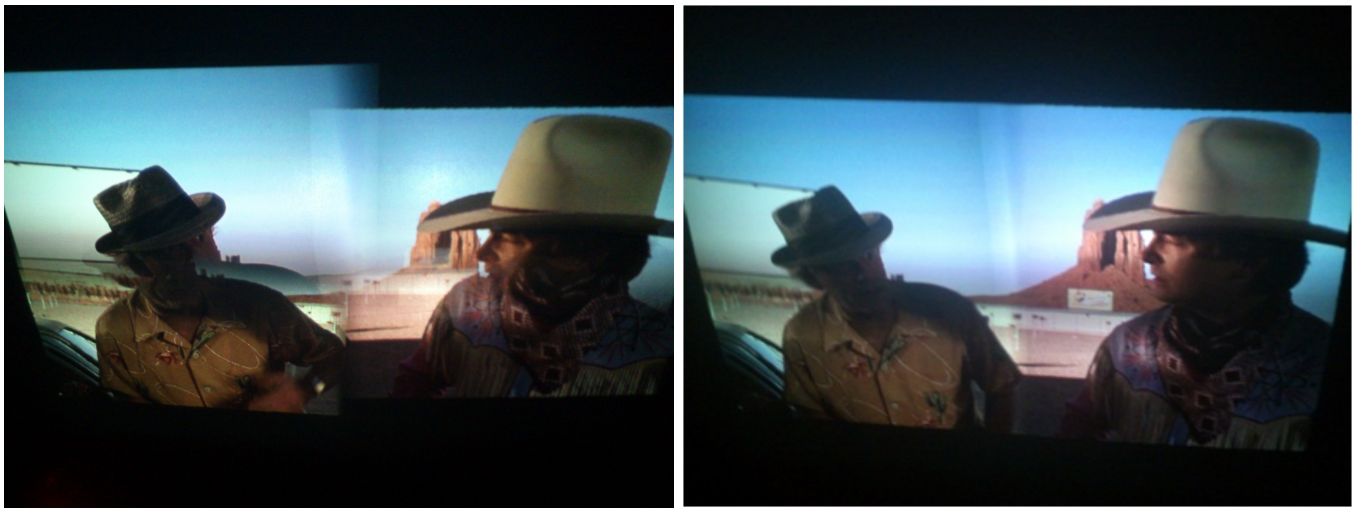

Fig. 7. A collaborative display made of 2×2 array of tiled mobile projectors, before (left) and after (right) registration.

compute the number of frame stalls L_i needed by device D_i to be synchronized with all other devices. Let us consider the frame j, $1 \leq j \leq m$, captured by the master camera. Let the decoded frame number for each D_i in frame j be denoted by fr_i^j. Let the fr_{min}^j denote the minimum of all fr_i^j in the frame j captured by the master camera. Let us consider the first frame captured by the master unit, i.e. $j = 1$. The frame stall L_i computed at the master device is given by

$$L_i = fr_i^1 - fr_{min}^1 \tag{1}$$

Note that using this, the synchronization error is less than one projection frame period ($\frac{1}{f}$). However, since the capture rate is m times more, we can achieve a synchronization error of less than $\frac{1}{mf}$ as described next.

For this, we use all the m images captured by the camera. Since we consider asynchronous projectors and cameras, all these images are taken within one projection frame period. Thus, the decoded frame numbers f_i^j for any device D_i can have at most two consecutive integer values with the transition between the two numbers happening at frame k_i. Thus,

$$fr_i^1 = fr_i^2 = ... = fr_i^{k_i} \tag{2}$$

$$fr_i^{(k_i+1)} = fr_i^{(k_i+2)} = ... = fr_i^m \tag{3}$$

The master unit first computes this k_i for each device D_i. For example, in Figure 9, $m = 7$ and $k_1 = 5$ for Projector 1. Then the master unit identifies the device with the maximum k_i as the most lagging and hence the reference unit to which everyone else should synchronize by stalling their frame display. The *subframe* delay S_i computed for each D_i is given by

$$S_i = \max_i k_i - k_i \tag{4}$$

Using this we can achieve an error of less than $\frac{1}{mf}$. For example, in Figure 9, Projector 1 is the lagging projector at the frame level, so projector 2 pauses 1 frame to wait for Projector one to catch up. At the sub-frame level, $\max_i k_i = 7$ (comparing to projector 2), and $k_1 = 5$. Therefore, we can derive $S_1 = 7 - 5 = 2$, which means $2/(7 \times f)$ sub-frame delay need to be applied to Projector 1 to achieve sub-frame synchronization.

B. Multi-Camera Based Distributed Synchronization

The synchronization algorithms we introduced so far is centralized and runs on a single master unit and uses the network to communicate the required frame pauses and sub-frame delays. Mobile network is often congested and unpredictable and this does not allow a short synchronization time. So, we next introduce a distributed approach for synchronizing the devices without using any communication over the mobile network. This approach assumes that every device has a camera looking at the display. Every device runs an identical SPMD code to achieve the synchronization in an entirely distributed manner using just the cameras for visual feedback and communication across each other. First we describe the method assuming a small set up of 2-4 projectors where the camera of each devices can see the entire projection area. Next we extend this method to a large scale system where each camera sees only its only projection and a little bit of its adjacent ones.

1) Small Scale Systems: In this distributed synchronization scheme, all devices capture images and adjust the frame stalls individually. That is, every device is now a master unit by itself. During the synchronizing period, each device decodes the image and calculates the required frame stalls S between itself and the reference unit. Then the device will stall S frames autonomously according to the calculated value.

After a pre-determined synchronization period, all projectors start the regular video playback, as shown in Figure 10. Red lines indicate the capture time of the cameras on each of the projectors. After calculating and applying the corresponding frame stalls, all the projecting devices are synchronized at frame 6.

2) Large Scale Systems: We have so far assumed that each camera can cover the entire projection area. However, this assumption may not hold when we gradually increase the

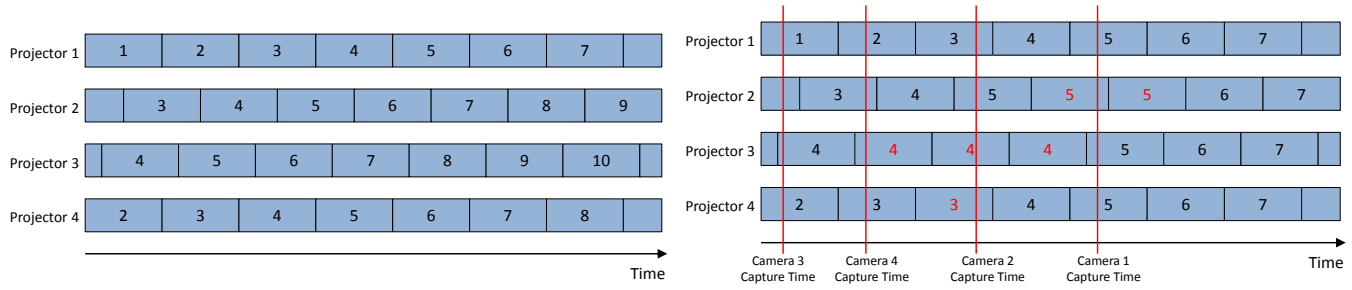

Fig. 10. Left: The frame displaying schedule before synchronization. Right: Devices synchronized distributedly.

number of devices. As the number of the projecting devices grows, the projection area also grows larger. Therefore, the devices may only be able to see part of the projection area, and will not be able to get the encoded information from out-of-sight devices. In this section, we alleviate this issue by synchronizing the devices locally, and achieve global synchronization after several rounds of information propagation in a tree-like fashion.

In order to build this tree structure, we assume all devices are tiled, and the camera on each device cannot cover the entire projection area, but is able to capture images covering its own projection area and at least 3 neighboring projection areas: on top of it, to the left of it and to the top-left of it(Figure 11).

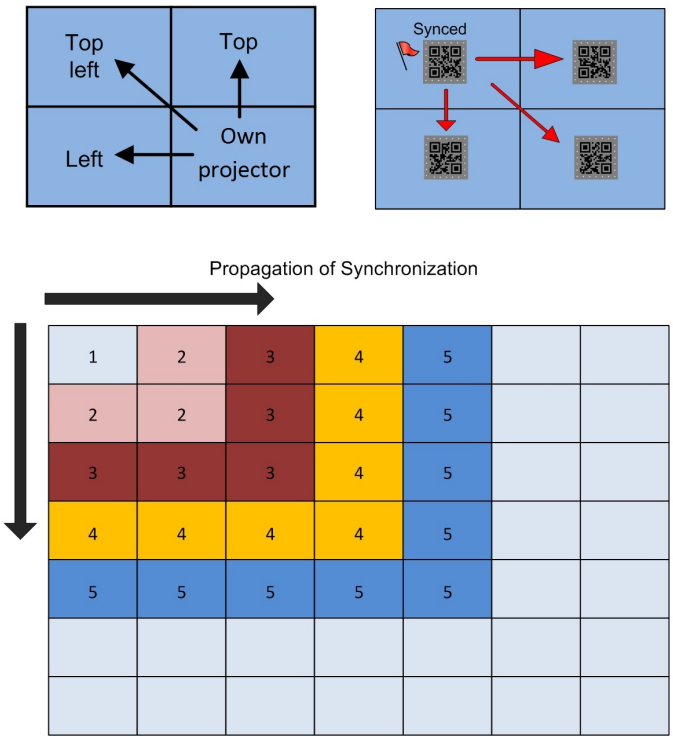

Fig. 11. Example of distributed synchronization. The numbers indicate the stage number based on which the particular device gets synchronized.

This scheme starts from the top-left corner of the tiled devices, using the most top-left device as the reference unit. The neighboring devices will first synchronize to this reference, and then can act as reference unit for other devices which are not yet synchronized.

We use a flag for each device to specify if it is synchronized and can be used as a reference unit for the neighboring devices. This flag is also embedded in the QR code projected by the device and it can be observed by the cameras of the neighboring projecting units. Therefore, those devices that find a nearby synchronized unit can synchronize themselves to the unit and update their flag to indicate that they are also synchronized. Thus, the synchronization starts with the top-left device which acts as the root node of the tree.

Consider a pair of neighboring devices, the maximum possible synchronization error e for such pair is equal to the granularity of the synchronization. This error e can propagate and accumulate through the path from the node leading to each device. The maximum length of this path, if each device synchronizes to a nearby reference in a $x \times y$ array of devices can result in the maximum accumulated error of $e((x-1)+(y-1))$ between the root and a device. In order to minimize the accumulated error, we desire to achieve the synchronization using the synchronization stages.

In each stage, devices only use those reference neighbors that are from a prior stage to perform synchronization. we use a stage indicator which is embedded in the QR code to indicate the stage number. Figure 11 shows the stage number in which each device gets synchronized with total number of devices which have achieved synchronization in the previous stages. The tree structure for propagation of synchronization. The devices in each stage for a level of the tree. However, this is slightly different than a traditional tree where each child can have multiple parents. Since each device can have multiple neighbors who have been synchronized in the previous step, they can synchronize using all these neighbors for greater robustness. Thus, all the neighbors that a device uses for synchronization is considered as its parents. Hence, each child in this tree has more than one parent. However, note that this does not affect the height of the tree which indicates the number of steps required to achieve complete synchronization. Note that this allows synchronization in the shortest number of stages equivalent to the height of the tree given by $max(x-1, y-1)$. Figure 12 shows a setup of 16 projectors with corresponding device IDs and the propagation path. Stage indicators are shown next to the tree. The red links indicate siblings i.e. the devices that has been synchronized in the same step. These are the edges which should be avoided to minimize the synchronization error from $e(x-1+y-1)$ to $e(max(x-1, y-1))$..

To obtain the stage indicator, we run another algorithm prior to the synchronization process. It starts by assigning 1 to the most top-left device, and propagates through the camera feedback. Each device would increase the minimum stage indicator by checking its captured image and use the decoded information to update its own stage number. To achieve better synchronization accuracy, sub-frame synchronization can also be adopted.

Note that in the previous algorithms, we detect the most lagging device among all of the devices, and synchronize them by stalling frames to let the most lagging device catch up. Since the reference unit here is not necessarily the most lagging device, the devices which are synchronizing to the reference unit may need to skip some synchronization sequences.

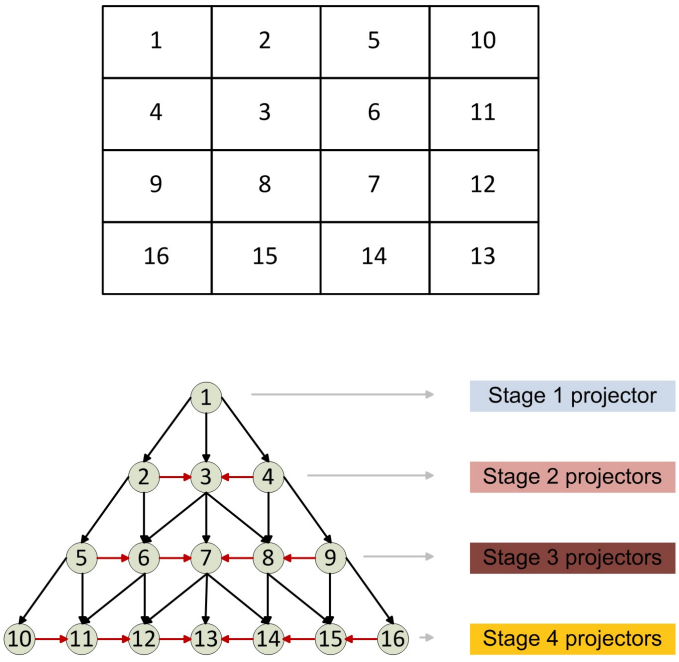

Fig. 12. The stage indicator and the propagation paths.

C. Re-synchronization During the Play

Although we assume the synchronization process only happens at the beginning of the video playback, in practice this may not be sufficient and we may need to re-synchronize when the video is being played due to the following reasons. First, when the network is congested, the streaming data might not arrive in time to be presented. Second, if the mobile device is running other programs, it may not have enough resource to decode the frames in time.

If we have only one device playing the video, we could pause the playback and wait until the frame arrives and decodes. However, in our tiled setup, pausing the video on one device may cause the others to run out of synchronization. Therefore, if pausing the video is not avoidable, the device should be able to notify others and let them know about the amount of frames that it is going to pause. Every other device will pause the same amount of frames to maintain the synchronization.

In order to implement the resynchronization, we continue to capture images after the video playback starts. If any projector has to pause, it will announce it by displaying a QR code containing the number of frame pauses needed. If all the devices see the whole projection area, every device adopts the maximum frame pauses needed from multiple captured QR codes. If the devices only sees parts of the entire projection area, the information again propagates in a tree like fashion as explained in the previous section.

IV. Superimposed Configuration

In the previous sections, the projecting frames only overlap near the edges to maximize the overall size and resolution of the frame. However, we can also adjust the frames to be superimposed on top of each other to have different benefits, in terms of brightness, frame rate, and power consumption. In this section, we describe such superimposed configurations, their installation and the various benefits resulting from such a setup.

One limitation of the current pico projectors is their low brightness (around 10 lumens). This can be alleviated by superimposing the same frame from multiple pico projectors and the overall brightness of the display can be linearly scaled. Similar to the tiled setup, in this case also, the frames displayed by each projector should also be spatially and temporally adjusted to get a seamless video. That is, we still need to use the visual feedback from the camera to obtain information for registration and synchronization from the QR codes.

In the previous tiled setting, the overlapping area for each projector is on the edges, so the QR codes projected by different projectors do not interfere with each other. But in the superimposed setting, the images are overlaid on top of each other, and so are the QR codes. Thus, superimposed QR codes from multiple devices prohibit the decoding of QR codes. Hence, we design a new algorithm that allows us to project the QR codes in distinct locations in such a manner that they do not clash with the code for another superimposing device.

Let us consider a superimposed setting with n devices, and each device is denoted by D_i, $1 \leq i \leq n$. The entire projection area is also divided into p, $p \geq n$, small areas which can contain a QR code without interfering each other and each small area is denoted by A_j, $1 \leq j \leq p$. Our algorithm is designed to assign each device D_i to a unique location A_j in a distributed manner. This is achieved by encoding one more parameter into the QR codes called `isLocked` that indicates if the projector has found and locked an area A_j that has not yet been occupied by other devices.

Initially, `isLocked` is set to false and each device D_i randomly selects an area number between 1 to p to project its QR code. Then each device captures an image of the whole projection area and attempts to decode it. Note that in a superimposed set up, each device can see the entire projection area by design. If the device is able to find and decode its own QR code, it will lock itself to that area by setting `isLocked` to true and broadcasts a message letting others know. If D_i discovers a clash with another device, it backs off and repeats

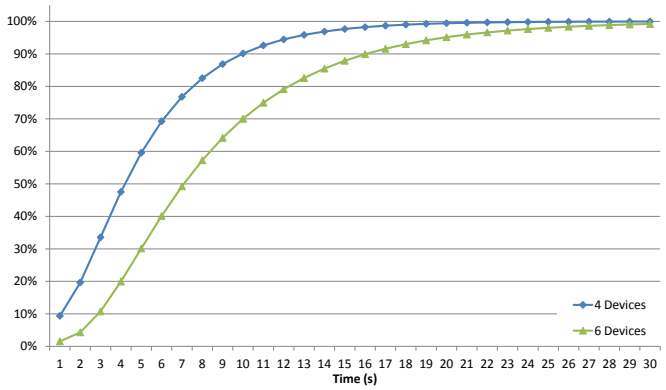

Fig. 13. Convergence rate of the QR code disambiguation algorithm

the same process until it finds a free spot. This scheme ensures that each device gets a unique spot to display the code. If a device is able to decode, it means no other device is projecting at the same spot and it is safe to lock itself to that position. Any other devices that try to project to the same position will not be able to successfully decode their QR codes. Also, if two devices select the same number very close in time, either both or one of them will back off. When all devices have their `isLocked` set to true, the registration and synchronization process can happen just as in the case of tiled devices.

Note that if we increase n, the number of devices, it also increases the convergence time for each device to find a unique spot. Figure 13 shows the simulation result of the convergence rate for the setup of 4 and 6 devices with 4 and 6 locations for QR codes respectively. After 30 iterations, the accumulated probability that every device is locked to a unique position is higher than 99%. Assuming that the projector and camera can operate at the speed of 30 fps, this process can converge within a second.

A. *Trading Off Different Resources*

As is evident, the superimposed configuration allows us to increase the brightness of the display. However, there are many other such opportunities where we may want to utilize other resources differently. For example, if power is more of an issue than brightness in a particular situation, we can reduce power consumption by superimposing n devices and letting them operate at a brightness scaled down by $\frac{1}{n}$. This may help extending the video play time for the users by a factor of n. In this section, we make first inroads to explore some such situations empirically and estimate roughly the amount of benefits such tradeoffs can bring.

1) Increasing frame rate: Consider the situation where we have each of n superimposing projectors to cycle through the video frames to be displayed. Thus, even though each projector runs at a lower frame rate, the quality of the display in terms of brightness in not compromised. For example, for a two projector set-up, one device will show the odd frames while the other shows the even ones. Thus each projector will run at 15 fps instead of 30 fps. the This, n devices each running at $\frac{30}{n}$ fps can create a 30 fps video. We refer to this setting

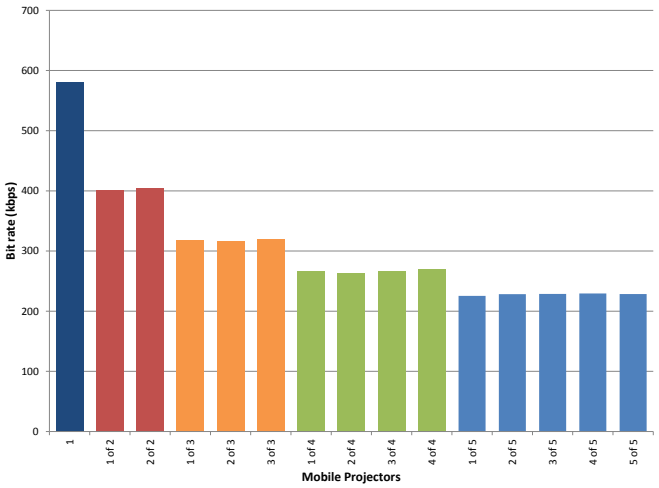

Fig. 14. Comparison of the required bit rate for a test video bitstream when using superimposed projectors with a single projector.

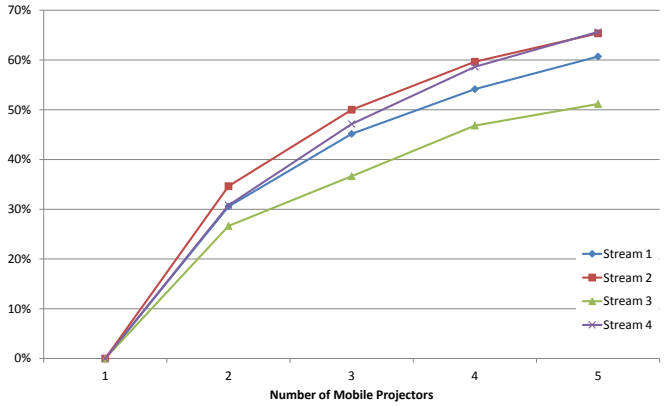

Fig. 15. Bit rate reduction per projector for 4 different video bitstreams when the number of projectors in a superimposed set up is varied.

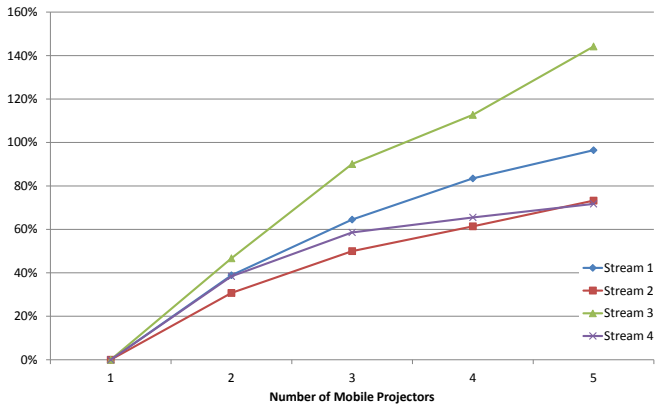

Fig. 16. Aggregated bandwidth increase for 4 different video bitstreams when the number of projectors in a superimposed set up is varied.

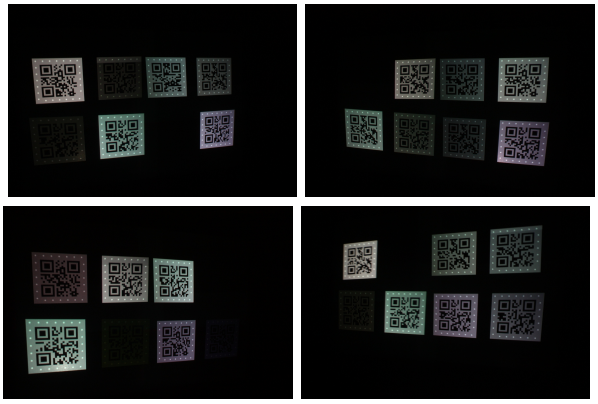

Fig. 17. The captured images by device 1, 2, 3, and 4 (in scanline order), in a 4-tiled setup.

as *alternating superimposed setup*. Let us now explore the consequence of such a set up.

First, since the video content delivered to each device is streamed at lower frame rate than a single full frame rate bit stream, the bandwidth requirement per projector is reduced compared to the bandwidth required for a single device receiving the whole video. Similarly, since each device runs at lower frame rate, the power consumed in the wireless receiver, the video decoder, and the projector of each device are reduced. Power savings per device in the collaborative setup allows us to extend the playback time of the video and enhance the user's viewing experience.

Fig 14 shows the bit rates compares the bitstream at 30 FPS for a single device with the bitrate at each device when in the setup of 2, 3, 4 and 5 alternating superimposed projectors are used to collaboratively display the content at 30 FPS. Note that since transcoding is involved the bit rate will not exactly scale down by n with a frame rate reduction by a factor of n. This is resulting from the lower performance of the video encoder as it encodes the alternating frames into separate bitstream. Alternating frames are less correlated comparing to the consecutive frames in the original video; therefore the temporal prediction used for coding these frames is less effective, resulting in lower compression performance. If the video is separated into more alternating video sequences (for more projectors), the temporal correlation between the frames of each separate video sequence is also decreased. Fig 15 shows the bit rate reduction per projector for 4 different video streams as we increase the number of the alternating projectors. While the bandwidth requirement is reduced per projector, the aggregated bandwidth for the whole system surpasses the requirement for the single bitstream video. Fig 16 shows this trend for the 4 video streams as the number of alternating projectors increases.

The above empirical data provides us a peek at all the different resource trade-off situations our collaborative display technology can bring forth. Though we do not explore this in depth or breadth in this paper, this will be the main focus of our future work.

V. Implementation and Practical Considerations

In this section, we demonstrate our synchronization algorithm using a 4-tiled projecting display. Our hardware setup is already shown in Figure 3.

We use the Texas Instruments DLP Pico Projector 2.0 Development Kits [12] as the projecting device, 3 MegaPixel Leopard Imaging Camera boards [3] as the camera module, and BeagleBoard-xM development boards [5] to control the projectors and cameras. We combine those tools as our prototype for our proposed algorithms, and demonstrate the ability to synchronize video playbacks between multiple devices.

It is important to mention that our tiled setup of projectors does not require precise positioning of the devices. Based on the camera feedback (Figure 17), we notify the user if the intended projection area has been covered by the camera. If the projected QR codes cannot be decoded, we will also notify the user that further repositioning is needed. After devices are well positioned, the registration program constructs the image with proper spatial positions and registers them using prior work in [6].

The pico projector used in our setup is able to project up to 2400 frames per second. However, the camera module's maximum capturing rate is only 15 frames per second, which is below the Nyquist rate of the video sample used in our system. Therefore, we incorporate the spatial repositioning of QR codes as shown in Figure 8. Figure 7 shows the snapshot of a well synchronized and geometrical registered video for the 4-projector systems. Based on the calculated homographies and image blending on the overlapping areas, each projector projects the video segment of size 640×480, creating a synchronized high resolution (1200×720) video.

In our implementation, highest synchronization accuracy demonstrated between each pair of directly synchronizing devices was 33 milliseconds. That results in 66 milliseconds accuracy in a pair of devices that are indirectly synchronizing. Considering our camera capture rate, pushing the synchronization accuracy to that level required displaying QR codes in 3 non-overlapping positions within a frame. Although we expect better accuracy with the actual faster cameras that are used in the current and the future mobile devices, our current results seem sufficient for the human visual persistence. Earlier works [7], [10] show that in most situations the users can only detect responses which lasts more than 100ms. Our synchronization accuracy is below this value, and we do not perceive any delay from our experiments.

We also implemented the superimposed configuration with two devices. Figure 18 shows a setup with two superimposed projectors before and after registration. Figure 19 also shows the two superimposed projectors displaying a grid before and after the registration. Each grid line is 2 pixels wide, and the total image size is 640×480. The brightness enhancement can be observed by comparing the non-overlapping area on the top-right corner of the image, while the rest of the image are overlaid on each other.

VI. Conclusion

We have introduced the concept of mobile collaborative video for the first time in this paper. We propose putting

Fig. 18. A two projector superimposed setup before (left) and after (right) registration.

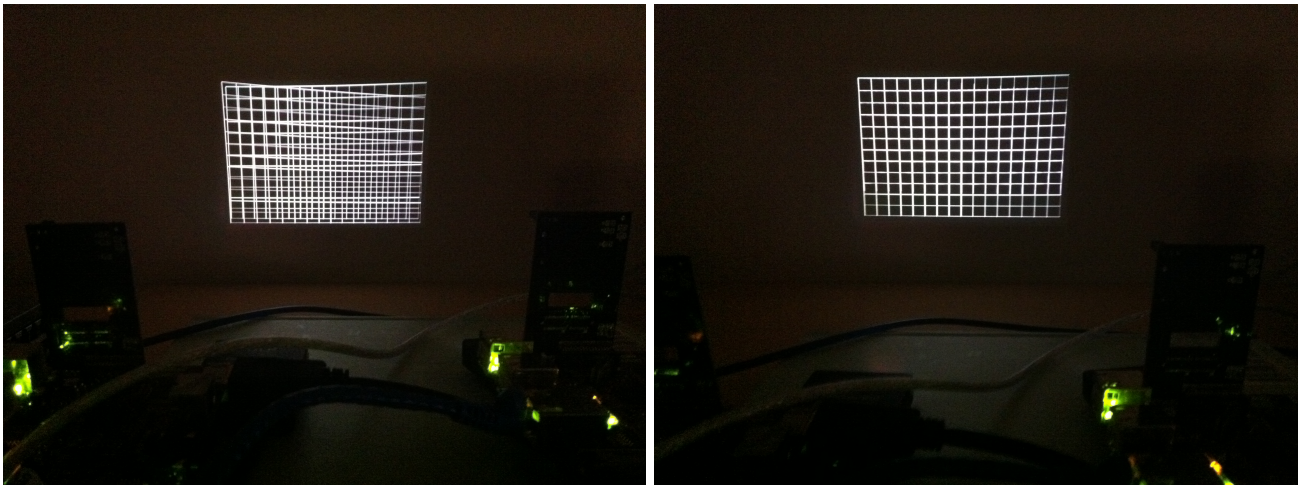

Fig. 19. A grid image before and after the registration in superimposed configuration

together multiple projector enabled mobile devices to deliver a single video to the user. Though registration techniques to achieve is available, the major roadblock turned out to be the temporal synchronization. Unlike wired network, wireless network is susceptible to delay and jitter due to unpredictable interferences making it unsuitable for achieving temporal synchronization. Therefore, we developed a visual feedback based synchronization scheme using the camera on the same device which is resilient to network conditions. We present both centralized and distributed scheme that can handle a range of different multi-device set up starting from a master coordinate tightly coupled scenario where each camera can see the entire projection area to an entirely distributed approach where each device can only see a part of the display. We can handle both tiled and superimposed projector configurations. Our method can achieve temporal synchronization at a subframe level. We have demonstrated this using prototype systems of both tiled and superimposed set ups using pico projectors augmented with BeagleBoard-xM development kit. Results demonstrate the success of our developed methodologies.

Collaborative mobile video offers a plethora of options for users in the future. It also provides several avenues to aggregate resources across devices while minimizing resource utilization per device. This has high-impact consequences in terms of power consumption and resource utilization. We provide a peek-preview of a few such example situations to provide an insight in to the exciting area we are starting to explore with this technology.

REFERENCES

[1] Canon PowerShot. http://en.wikipedia.org/wiki/Canon_PowerShot_A.

[2] Casio Exilim. http://en.wikipedia.org/wiki/Casio_Exilim.

[3] Leopard 3M Camera Board. https://www.leopardimaging.com/.

[4] QR Code. http://www.qrcode.com/en/aboutqr.html.

[5] BeagleBoard-xM. http://www.beagleboard.org.

[6] E. Bhasker, P. Sinha, and A. Majumder. Asynchronous distributed calibration for scalable reconfigurable multi-projector displays. *IEEE Transactions on Visualization and Computer Graphics (Visualization)*, 2006.

[7] P. A. Laplante. *Real-Time Systems Design and Analysis*. IEEE Press 2nd edition, 1997.

[8] Pico projector news and resources. http://www.dqmagazine.com/news/singapore-develops-smartphone-pico-projectors-10150/.

[9] Pablo Roman, Maxim Lazarov, and Aditi Majumder. A scalable distributed paradigm for multi-user interaction with tiled rear projection display walls. *IEEE Transactions on Visualization and Computer Graphics*, 2010.

[10] B. Shneiderman. Response time and display rate in human performance with computers. *Computing Surveys*, 16(3):265–285, 1984.

[11] P. Sinha, E. Bhasker, and A. Majumder. Mobile displays via distributed networked projector camera pairs. *Projector Camera Systems Workshop*, 2006.

[12] Texas Instruments. DLP Pico Projector Development Kit. http://www.dlp.com/pico-projector/.