Magnifying Smartphone Screen Using Google Glass for Low-Vision Users

Shrinivas Pundlik, HuaQi Yi, Rui Liu, Eli Peli, and Gang Luo

***Abstract*—Magnification is a key accessibility feature used by low-vision smartphone users. However, small screen size can lead to loss of context and make interaction with magnified displays challenging. We hypothesize that controlling the viewport with head motion can be natural and help in gaining access to magnified displays. We implement this idea using a Google Glass that displays the magnified smartphone screenshots received in real time via Bluetooth. Instead of navigating with touch gestures on the magnified smartphone display, the users can view different screen locations by rotating their head, and remotely interacting with the smartphone. It is equivalent to looking at a large virtual image through a head contingent viewing port, in this case, the Glass display with $\sim 15^\circ$ field of view. The system can transfer seven screenshots per second at $8\times$ magnification, sufficient for tasks where the display content does not change rapidly. A pilot evaluation of this approach was conducted with eight normally sighted and four visually impaired subjects performing assigned tasks using calculator and music player apps. Results showed that performance in the calculation task was faster with the Glass than with the phone's built-in screen zoom. We conclude that head contingent scanning control can be beneficial in navigating magnified small smartphone displays, at least for tasks involving familiar content layout.**

***Index Terms*—Google Glass, low-vision aid, screen magnification, smartphone app.**

I. Introduction

Loss of visual acuity (VA), caused by various conditions such as age related macular degeneration (AMD) or optic nerve atrophy, leads to difficulty in reading and discerning fine details. Magnification is the most effective method of compensating for such visual loss. Magnification devices help people with central vision loss perform routine daily tasks such as reading [1]–[3]. Various reading assistance devices, such as head-worn and hand-held optical magnifiers, and electronic magnifiers are commercially available. With the advent of personal computers, screen magnification approaches were explored [4], [5] and many screen magnification software programs are available [6]–[8]. Modern operating systems for desktop computers or notebooks provide built-in magnification features [9], [10]. As portable mobile electronic displays become more prevalent, the digital magnification of the screen will be more frequently used by a low vision population.

Along with the general population, people with low-vision are becoming smartphone users. Some surveys have found that the prevalence of smartphone use among people with low-vision is not different from the general population [11], [12], especially in developed countries. Popular mobile operating systems such as iOS and Android have built-in accessibility features for blind and visually impaired users. For people with residual vision using smartphones, magnification is the most widely used accessibility feature [12], [13].

One of the major difficulties in working with small phone screens at higher magnification levels is the loss of context, which can make screen navigation difficult. Magnification in smartphone displays is typically controlled with a pinch action and navigation is achieved by panning using scrolling gestures (dragging fingers over the display), similar to a zoom + pan interface [14]. Screen size is an important factor affecting user performance in the zoom + pan interface [14]. Specifically, consider an Android smartphone with a screen size of 13 $\times$ 6.8 cm that can offer a maximum magnification of 5.2 $\times$ with the built-in screen zoom. At the maximum screen zoom, a 1 cm $\times$ 1 cm icon covers about 33% of the screen area. Comparatively, a 1 cm $\times$ 1 cm icon would cover only about 3% of the total screen area for the same magnification on a 21 in monitor. Stated another way, at 5 $\times$ magnification, only 4% of the original display is available on the screen at any time. As a result, navigating a magnified screen on a smartphone could be very time-consuming for low-vision users.

Previous visualization research found that physical navigation (movement of body/head) can reduce (improve) performance time compared to purely virtual navigation (such as zoom + pan interface) [15], [16]. Here we present a novel way to visualize and interact with smartphones using a head mounted display (HMD–Google Glass [17]) for easy accessibility by low-vision users. We developed a Google Glass screen sharing app that projects the screen of a paired smartphone onto the Google Glass display at an appropriate magnification. The user, wearing the Google Glass display, can view a zoomed-in smartphone screen, and pan by moving his or her head. They can also interact with the smartphone using the Google Glass swipe side panel. The effect can be likened to the user looking

Manuscript received June 08, 2015; revised December 12, 2015, and March 07, 2016; accepted March 10, 2016. Date of publication March 23, 2016; date of current version January 06, 2017. This work was supported in part by an unrestricted gift from Google, Inc. to E. Peli, and by the Eleanor & Miles Shore Fellowship award to G. Luo.

S. Pundlik, R. Liu, E. Peli, and G. Luo are with the Schepens Eye Research Institute, Mass Eye and Ear, Harvard Medical School, Boston, MA 02114 USA (e-mail: shrinivas_pundlik@ meei.harvard.edu).

H. Yi is with the Computer Science Department, Northeastern University, Boston, MA 77005 USA.

Digital Object Identifier 10.1109/TNSRE.2016.2546062

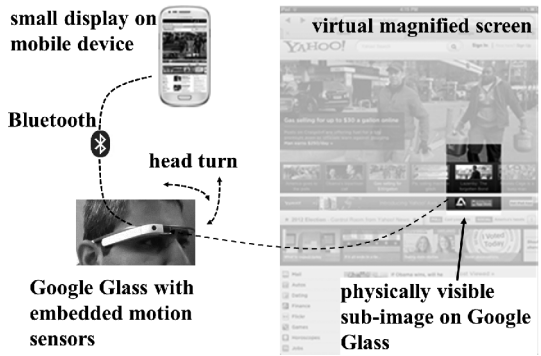

Fig. 1. Schematic of the Google Glass screen share system. Google Glass presents a magnified sub-image of the smart phone screen. Head position, sensed by motion sensors in Google Glass, controls the portion of the smartphone screen to be displayed.

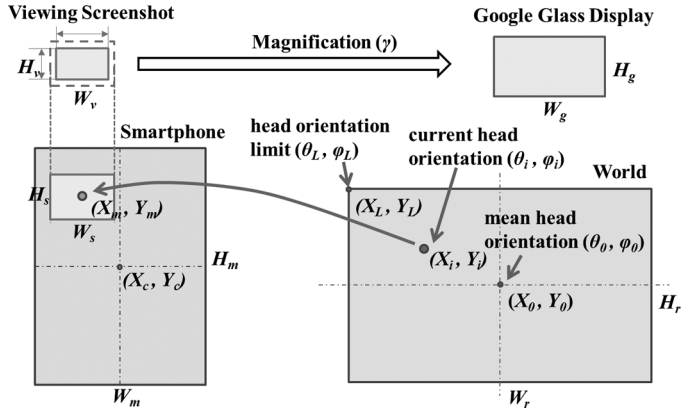

Fig. 2. Mapping the smartphone screen with Google Glass display. Head movement of the user defines a virtual plane in the world, the size of which is set during the system calibration step. At a given instant, Google Glass sends the orientation of the head which is mapped to a location on the screen of the smartphone. This is the center of the viewing window to be transferred to the Google Glass display and its size is determined by the magnification factor γ.

at a large virtual image of the phone's screen through a viewing port, the Glass display ($\sim 15^\circ$ diagonal field of view). Based on visual angle, the size of the Glass display is comparable to that of a typical smartphone held at about 35 cm away (for example, the angular size of the 8.7 cm x 5.3 cm display of Samsung Galaxy S3 mini smartphone is $\sim 16^\circ$). At similar magnification levels, the two displays can be considered equivalent, and hence viewing content on the Google Glass display is not significantly different for a user than viewing it on a phone display.

Our approach is different from gaze-controlled video magnifiers [18], [19] in multiple ways. Panning control based on head position is independent of eye movement, which is heavily used in reading and viewing. Thus, the users will be provided with relatively stable images. Also, head position is relatively easier and more robust to measure than gaze position, and therefore more suitable for controlling the display orientation in our application. The availability of integrated motion sensors on the Google Glass make head tracking relatively straightforward to implement. The following sections of the paper introduce the concept of head motion navigation of magnified smartphone screens, describe the technical details of implementing this concept using Google Glass, present the results of a preliminary evaluation study to test the benefit of the approach, and discuss its limitations and future potential.

II. Head Controlled Screen Zoom

The Google Glass screen share system maps the smartphone screen to a larger virtual space in front of the user and shows a section of it (depending upon the magnification) on the Glass display (Fig. 1). The location of the viewing window (Glass display) depends on head orientation. For example, a person looking straight ahead will see the center portion of the smartphone screen on the Glass display. As the person turns their head in the vertical direction, the viewing window will move correspondingly to display the upper area of the smartphone screen. As the user is intuitively aware of which portion of the virtual screen is being viewed using proprioperceptive feedback from the head, the system helps the user to maintain orientation and ease navigation.

Fig. 2 shows the mapping between the mobile phone and Google Glass (not drawn to scale). The system transfers only a portion of the smartphone screen at a time to the Google Glass display ($W_g \times H_g$), to minimize the processing time. The Google Glass display moves as the user's head turns in space, and this movement is tracked by reading the azimuth (θ) and pitch (φ) angular values available from the built-in orientation sensors in the Glass. The roll angle of the Glass is not considered here as it does not affect the overall display system. Assuming that head turns are limited by a reasonable angular range in either direction, we can constrain the overall motion of the Google Glass display to a virtual rectangle in the world of dimensions $W_r \times H_r$ centered at (X_0, Y_0). The point (X_0, Y_0) corresponds to the neutral head orientation given by the azimuth and pitch angles of θ_0 and φ_0, respectively. The top-left angular limit of head movement is given by (θ_L, φ_L), and corresponds to the upper left corner of the virtual rectangle. The neutral head orientation and the maximum allowable orientation in the horizontal and vertical direction can be defined/calibrated by users beforehand. The range of calibrated head movements to map the display in both directions can be different, and affects the speed with which the viewing window updates with the head movement: narrow range means faster shifts, whereas wider range means slower changes in viewing window locations.

We now need to know the location and the size of the viewing window on the smartphone screen that needs to be mapped to the Google Glass display. The location of the viewing window to be extracted from the smartphone screen is determined by the current head orientation, and its size is determined by the selected magnification level. Let the current head orientation be (θ_i, φ_i) corresponding to the location (X_i, Y_i) in the virtual screen, and let $W_m \times H_m$ be the smartphone screen dimensions, then the location of the center of the viewing window on the smartphone screen, (X_m, Y_m), is given by

$$X_m = \frac{(X_0 - X_i)}{(X_0 - X_L)}\left(\frac{W_m}{2}\right) = \frac{(\theta_0 - \theta_i)}{(\theta_0 - \theta_L)}\left(\frac{W_m}{2}\right),$$
$$Y_m = \frac{(Y_0 - Y_i)}{(Y_0 - Y_L)}\left(\frac{H_m}{2}\right) = \frac{(\varphi_0 - \varphi_i)}{(\varphi_0 - \varphi_L)}\left(\frac{H_m}{2}\right).$$

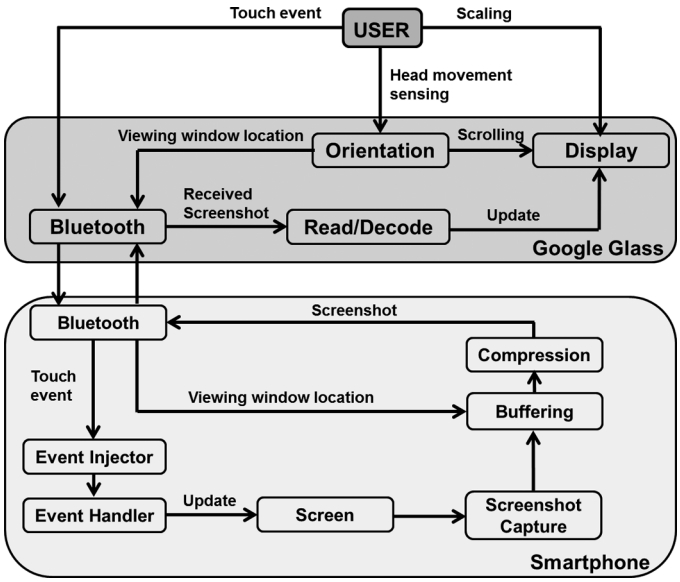

Fig. 3. System overview. Smartphone screen share system has two separate components: one running on the Google Glass (client) and the other running on the smartphone (host). Client receives, decodes, and displays the smartphone screenshot. It also sends the viewing window location and any user generated events to the host. Host captures the screenshots, compresses them, and sends them to the client. It also manages the received user events. Communication between the two parts takes place via Bluetooth.

Since (θ_0, φ_0) and (θ_L, φ_L) are set during the calibration process, and W_m and H_m are known, the center of the viewing window can be calculated based on the current head orientation values of (θ_i, φ_i). If γ is the current magnification value ($\gamma \geq 1$), then the size of the screenshot is scaled according to

$$W_v = \frac{W_g}{\gamma}, \qquad H_v = \frac{H_g}{\gamma}$$

so as to fit the Google Glass display. The actual dimensions of the viewing screenshot transferred from the smartphone are $W_s = W_v + \delta$ and $H_s = H_v + \delta$, where δ is a small buffer added to the viewing window dimensions so that an empty border zone is not displayed when the head moves. The Google Glass transfers the current head position and magnification values to the paired smartphone to determine the appropriate sub-image to be transferred back to the Google Glass.

The current implementation of the screen sharing app locks the smartphone display in portrait orientation, because it is suitable for the two apps used in our evaluation experiment. Landscape orientation can be easily implemented for applications that are more suitable to be used in that mode.

III. System Description

Fig. 3 shows the detailed block diagram of the system. The screen share app consists of two separate applications: a host app running on a smartphone, and a client app running on the Google Glass. The host and the client apps communicate via Bluetooth. Initially, the host sends the entire screenshot to the client. The client determines the appropriate sub-window to display based on the current head position of the user. The client sends the current viewing window location and the user generated events to the host, based on which, the host sends back either a part of or the entire compressed screenshot of the current screen to the client. Upon receiving the corresponding information, both the host and the client update their states in a cyclic manner. The Google Glass either replaces the existing screenshot with the newly received screenshot, or merges the changes to the viewing window in the display buffer.

A. Hardware Description

The proposed system was implemented and optimized for Google Glass and the Samsung Galaxy S3 mini smartphone. The Google Glass is a head mounted wearable mobile platform. It has an optical see-through display with a resolution of 640 $\times$ 360 pixels, covering a visual angle of $13^\circ \times 7.3^\circ$ for the right eye. The optical display assembly is situated above the primary line of sight, requiring users to look up to see the display. The virtually displayed image is formed about 8 feet away from the viewing eye. The OMAP3 based processor runs Android 4.4 (KitKat) that supports Bluetooth connectivity and a motion sensor (tri-axial gyroscope, accelerometer, and magnetometer that can be used to obtain the orientation information), among many other capabilities. A touch sensitive side panel can recognize different touch gestures: horizontal swipe, vertical swipe, short tap, and long tap. Bone conduction speakers provide the audio input to the users. The Samsung Galaxy S3 mini smartphone has a 4-in diagonal screen, with 800 $\times$ 480 pixel display. It runs Android 4.1 (Jelly Bean) on a 1 GHz dual core Cortex A9 processor. It supports Bluetooth v4.0 connectivity for communication with Google Glass.

B. Host Side Processing (On a Smartphone)

The main processing blocks on the host side are: 1) taking a screenshot, 2) preparing the screenshot image for transfer, and 3) handling the user events received from the client.

1) Capturing a Screenshot: Capturing a screenshot is the most time consuming step. The Android graphics stack primarily consists of image stream producers (like the media player or camera preview, OpenGL ES etc.) that produce buffer data, and image stream consumers (typically SurfaceFlinger system service) that are responsible for preparing the buffer data to be displayed and sending it to the hardware abstract layer (HAL). To capture the screenshot, the data has to be accessed either at the producer level, at the buffer stage, or at the consumer level (before it reaches HAL). We explored four approaches for screenshot capture: using the Android SDK function *getRootView()*, using the system command of "*screencap*", directly accessing the *SurfaceFlinger*, and accessing the frame buffer */dev/graphics/fb0*. Using the *getRootView()* SDK function, we can only capture the screenshot of the app itself, which makes it unusable for our application. Obtaining the screenshot via the *screencap* command is similar to capturing it using device hotkeys, such as simultaneously pressing the home and volume buttons. This is a clean and robust method in which the command is issued by invoking the shell. However, this method is very slow, because the captured screenshot is written to the external storage and read back (taking about 500 ms for capturing an 800 $\times$ 480 blank screen). Although efficient, the screen capture using the *SurfaceFlinger* service is challenging because its API is

hidden. Any modifications to the *SurfaceFlinger* would require rebuilding the Android system from the source. The selected option of capturing the frame buffer *(/dev/graphics/fb0*) is a compromise between feasibility and speed.

2) Preparing the Screenshot for Transfer: The captured screenshot needs to be compressed to enable efficient transfer over the Bluetooth channel. Since the average bandwidth over Bluetooth is about 200 KB/s, an uncompressed bitmap of size 800×480 would take about 8 s to be transferred to the Glass, which would be unreasonable. The screenshot is compressed using JPEG compression with a quality factor of 75. It is a compromise between keeping the screenshot quality high while reducing the file size. Even after compression, sending the whole smartphone screenshot every time can be inefficient (on the order of a few hundred milliseconds). For many reasons, transferring the whole screenshot may not be necessary. When the display magnification (on Google Glass) is high, only a small fraction of the smartphone screen is visible to the user. For many screens, when users just scan the content without interacting with the smartphone, the displayed content usually changes only in small local areas, e.g., clock in the corner, and therefore the full screenshot does not have to be transferred again. If only a part of the screen is being transferred, the frame rate and latency can be greatly improved.

Using the location and size of the current viewing window received from the Glass, the phone screenshot is cropped corresponding to the sub-image to be shown in the Glass along with an additional buffer area, which is used to allow a level of scanning/ panning when the head moves without updating the transferred image (see Fig. 2). The cropped screenshot can be sent to the Glass much more efficiently because of its small size. The problem with this approach is that when users move their head too fast, they may see a portion of the old screenshot, as the Glass has not received an updated screenshot for the viewing window area. Increasing the buffer area can alleviate this possibility, but as mentioned earlier, at the cost of a slower frame refresh rate.

To address the image transfer limitation and to improve the user experience, we developed a strategy that combines both full and cropped screenshot transfer methods (Fig. 4). Once the latest screenshot is available, the viewing area of the current and the previous screenshot are compared to check whether they are different. If a change is detected in the viewing window, it needs to be updated with the highest priority, and therefore only the current viewing window is sent. If the viewing window does not change but there is a change somewhere else, it indicates that the user is viewing a locally unchanged section and therefore will not notice a slower frame rate caused by a whole screen update. In this situation, the entire screenshot is sent to the Glass. If there is no change in the whole screen, then there is no need to send a screenshot.

Byte wise comparison of successive screenshots is computationally inefficient. It may take about 80 ms on the Galaxy S3 mini to compare 800×480 resolution images. Direct comparison of bitmaps using an optimized method in Android SDK is far more efficient, taking only about 10 ms. However, it only returns a binary output about whether the two bitmaps being compared are different or not. If no change is detected in the viewing area, then the entire screenshot is subject to the change detection function. Hence, the change detection step is called twice in certain situations, as shown in Fig. 4. The overall time required to transfer the screenshot can vary greatly and depends on the Bluetooth bandwidth, the operational environment, and the size of the screenshot.

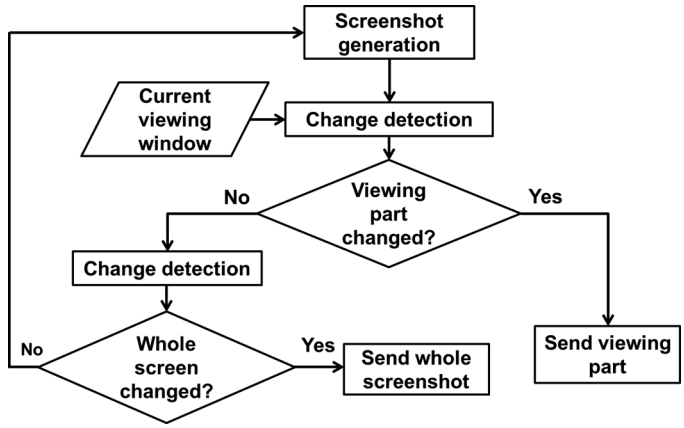

Fig. 4. Strategy for sending the screenshot for faster performance. Change detection in the viewing part versus the whole screen is used for keeping the size of the transferred screenshot to a minimum in order to improve overall system speed.

3) Handling Touch Events: The smartphone receives the users' current head orientation and the level of display magnification from the Glass. This information is used by the smartphone to compute the size and location of the current viewing window, which is necessary for determining the screenshot to transfer. To facilitate interaction with the smartphone while using Google Glass without the need to look at the phone screen directly, the smartphone also receives touch events sent by the Glass (user tapping or swiping on the side touch panel). The received events are injected in the smartphone, where they simulate corresponding motion events such as tapping on the viewed screen section. These events are then handled by the mobile phone as regular touch events and may result in changes in the current smartphone screen.

C. Client Side Processing (Google Glass)

The main processing steps on the client side are: 1) orientation sensing, 2) reading and decoding the received screenshots, 3) displaying the image with correct scaling and position shift, and 4) facilitating interaction with the smartphone screen. The monitoring of head rotation and its use in the smartphone is described above.

1) Orientation Sensing: Built-in motion sensors in the Google Glass are used to determine the orientation of the device in world coordinates. In our app, only the azimuth and pitch are used to compute the head orientation. Fig. 5 shows the axes of the orientation for the Google Glass display. Azimuth is the angle between magnetic north and the z-axis (rotation around the y axis). The range of azimuth is from 0° to 360°. Pitch is the rotation of the Glass around the x-axis. The range of pitch angle varies from -180° to 180°. These two angles describe the user's head orientation and are used to compute

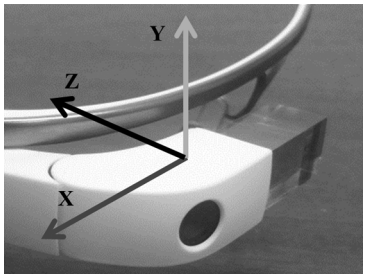

Fig. 5. Google Glass motion sensor axes. Azimuth (rotation around the y-axis) and pitch (rotation about x-axis) angles describe the 2-D head orientation, which in turn are used to compute the coordinates of the viewing window on the smartphone screen.

the coordinates of the viewing window to be mapped onto the display (rotation along the z axis is not considered).

For calibration, the user aligns the current or any other desired head orientation to the center of the smartphone display using a long-press gesture on the Glass side touch-panel. This is set as the neutral or reference orientation until it is reset when the user repeats the long-press gesture. After setting the reference or central orientation, the user can also reset the head movement range that maps to the smartphone display size. The top-left limit of the head movement is defined by the user by moving the head in that direction and double tapping the touchpad. The orientation values for top-left limit and the center are sufficient to calculate the dimensions of the rectangle in the world coordinates (symmetric around the central orientation), to which the smartphone display is mapped. Based on the magnification value selected by the user, and the current orientation of the head, the appropriate viewing window is presented on the display. The azimuth values loop back when the head rotates across the measurement bounds of the device (going from 360° to 0° or vice versa), which can cause erratic shifts in the display. The solution is to count the turns of the Glass. Whenever the absolute difference from the previous azimuth value is larger than 180°, then we add or subtract one turn to the counter (n) and the current azimuth value is updated as

$$\theta_i = \theta_i + (n \times 360)\,, \quad n+ = \begin{cases} -1, & \theta_i - \theta_{i-1} < 180 \\ 1, & \theta_i - \theta_{i-1} > 180 \\ 0, & \text{otherwise} \end{cases}$$

where θ_i and θ_{i-1} are the current and the previous azimuth values (measured in degrees), respectively. By doing so, the value of azimuth becomes continuous and the display transitions smoothly with head movements in the world.

2) Reading and Decoding Screenshots: The screenshots received by the Glass through Bluetooth are written to an image buffer and decoded. These are either the entire screenshots or portions of the screenshots (corresponding to the head rotation and the magnification level). While reading and decoding operations can be performed in separate threads on the Google Glass, we observed that it was not faster than a simple buffer with sequential reading and decoding operations. A possible reason could be that there were already many threads running on the Glass and switching between the threads was time consuming. Hence, using a sequential reading/decoding operation provided better overall system performance.

3) Displaying the Screenshot: Three inputs are fed into the Glass display module: the received screenshot, the head orientation, and the magnification level selected by the user (via the Glass touchpad). An image buffer equal in size to the maximum screenshot size is created and initialized to zero. Using the head orientation, the center of the viewing window is computed. As an updated screenshot is received, it is determined whether it is a full or partial frame. If a partial screenshot frame is received, then only the corresponding image block of the image buffer is updated. Otherwise, the entire image buffer is overwritten with the updated screenshot. The size of the viewing window to be mapped on the display is determined by the magnification level (there is an inverse relationship between them). Using the computed size and location of the viewing window, the appropriate portion of the image buffer is mapped to the Glass display.

4) Remote Interaction With Smartphone: Being able to remotely interact with the smartphone is a key feature of our system. It would be very difficult for the users to click or tap on the smartphone screen while looking at the Glass display because they would need to know where their fingers are actually touching on the screen. It would not help to show the current touch area of the smartphone screen on the Glass, because at that time the touch event on the smartphone has already taken effect and it might have been outside the Glass view. Our solution is to let the user interact with the smartphone screen through the Glass. A cursor is shown at the center of the Glass display (a red circular dot in the present version of the app). The Glass handles the tap event on the touchpad by sending the coordinates of the current cursor location (X_m, Y_m in Fig. 2) to the smartphone device so that it can simulate a touch event at that position. Thus, tapping on the Glass would be equivalent to touching the smartphone screen. The user can precisely control the location of the touch event by moving their head to aim the cursor at the desired screen location at any magnification level. While the current version only recognizes the tap gesture, other gestures can be implemented using the Glass touchpad.

IV. System Performance Evaluation

We examined the impact of screen type and magnification level on performance timing operations (reading and decoding) on the host and client. Time required for screenshot compression on the smartphone, and for reading and decoding it on the Google Glass depends on the type of screen being manipulated and the magnification level. Screenshot capture is independent of the magnification level and for a given magnification level it can be severely affected by the system load at run-time, which is unpredictable.

In the first set of experiments, we measured the time taken by the screenshot compression, reading, and decoding modules for five different types of captured screens: the host app, calculator, music player, map, and webpage (Fig. 6). These five screens were chosen to represent the variety of content displayed on smartphones. The compression time at different levels of magnification (1 ×, 2 ×, 4 ×, and 8 ×) was recorded on the smartphone [Fig. 7(a)], whereas the reading and decoding time was recorded on Google Glass [Fig. 7(b) and (c)]. Screens with rich content such as the webpage and map required more time to compress compared to the host app screen or music player.

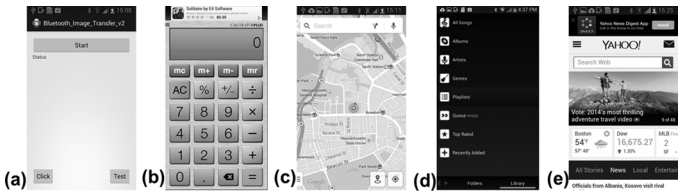

Fig. 6. Screenshots of the apps used for measuring the performance of our screen share system: (a) host app, (b) calculator, (c) map, (d) music player, and (e) web page.

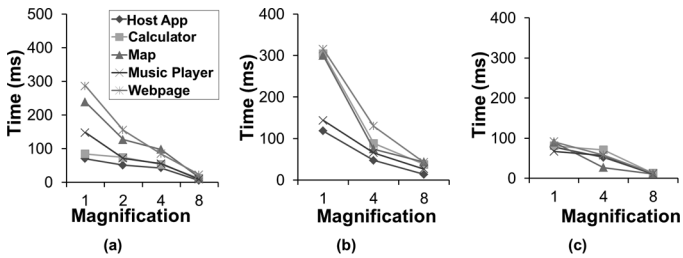

Fig. 7. Time required to compress, read, and decode five types of smartphone screens at different magnification levels. Reading the screenshot on Google Glass was more time consuming than decoding the screenshot. Overall, the content of the screen seemed to be the main factor in determining the time required by these processes on the smartphone and the Glass. Screens rich in content, like the map or webpage, in general took longer than relatively sparse screens like the host app. (a) compression time (smartphone). (b) reading time (Google Glass). (c) decoding time (Google glass).

The compression time reduced steadily with a higher magnification level for all five screen types because the size of the transferred viewing window is reduced with higher magnification. For a given screen type, reading time was usually longer than decoding time, especially at lower magnification levels. The calculator, map, and webpage required a longer reading time than the music player and host app. However, the decoding time was similar for all screen types. The reading operation was a buffering operation (byte wise data copying) whose performance was directly related to the data size.

In the second set of experiments, we measured the time required by the various modules of the system for static and dynamic screens that displayed a map. The Map app was chosen for this test. The data for these experiments were collected by: 1) starting screencast, 2) opening the Map app on the smartphone, and 3) navigating through the screen at different magnification levels. For simulating dynamic screens, the map was moved manually on the phone by an operator searching for locations. Fig. 8(a) shows the screenshot compression time for static and dynamic scenes. Overall, the time required to process static screens on the smartphone was much shorter compared to dynamic screens because the content of dynamic screens changed rapidly over time, which resulted in a larger data size after compression. The time required to read and decode the screenshots on the Google Glass was similar in both static and dynamic cases [Fig. 8(b) and (c)]. Finally, the overall frame rate of the system as a function of magnification is shown in Fig. 9. The frame rate is calculated based on the number of frames received by the Google Glass for a static map screen. The frame rate includes the time required to capture the screenshot and transfer it via Bluetooth.

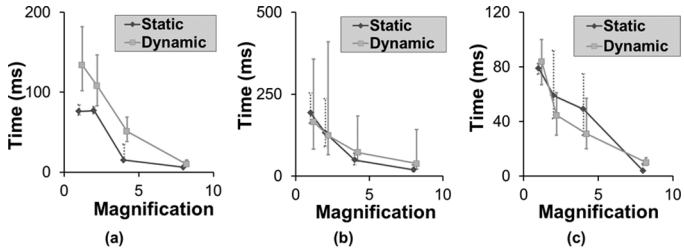

Fig. 8. Comparison of timing performance between static and dynamic screens of the Map app on the smartphone and Google Glass for different processing steps. The plot shows median time over about 200 frame transfers recorded for different magnification levels, with the error bars representing the inter-quartile range. Dynamic screens required longer time for capture and compression. There was not much difference between the time required to read and decode the static and dynamic screens on the Glass. (a) Compression time (smartphone). (b) Reading time (Google Glass). (c) Decoding time (Google glass).

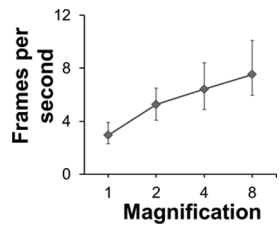

Fig. 9. Median system frame rate (measured as number of incoming screenshots for the map app) for different magnification levels recorded over about 200 frame transfers. Predictably, the frame rate increases with the magnification as the effective size of the screenshot reduces. The error bars represent the inter-quartile range.

Fig. 10. Screenshots of different screens of the Poweramp music player used in the music playing task. Options denoted by a solid gray box in (a), (b) and (c) lead to subsequent screens, whereas options denoted by a dashed gray box in (b), (c), and (d) take the user to previous screens. Icon highlighted by the dotted box at the top in (d) leads back to the main menu. (a) Main menu contains various items, but only the playlists option was used for this task. (b) All the stored playlists, with one of the playlists selected. (c) List of all the songs in the selected playlist. (d) Media player playing the selected song. Highlighted items (rectangles) are shown for the sake of illustration only and were not present on the screen during the experiment. (a) Main menu. (b) Playlists. (c) Song list. (d) Media player.

V. User Evaluation

A pilot study was conducted to evaluate the proposed approach while performing two routine real world tasks. Specifically, we compared the performance of subjects on the same tasks in two conditions: without Google Glass (using the smartphone with the built-in screen zoom feature) and with the Google Glass. Task performance was measured in terms of the time to complete the task. The tasks chosen for this study were: performing multiplication operations using a calculator app and

playing requested songs using a music player app. Performance was compared within subjects.

A. Methods

When performing the tasks using the smartphone screen directly, screen zoom was set to $8\times$. When using the Google Glass, the screen magnification was adjusted in such a way that the visible area on the Glass display was the same as in the magnified smartphone screen. This resulted in the same angular size of characters when viewed on the Glass and on the magnified smartphone viewed at a 25 cm distance.

The calculation task was performed with the Calculator Plus app (free on Google Play) [20] [see Fig. 6(b)]. The task consisted of performing a series of multiplication operations involving two two-digit numbers. There were 12 trials with each device, one operation per trial. The multiplication operations were randomly generated and were the same for all subjects. At the beginning of each trial, the experimenter read out the two numbers, and after obtaining confirmation from the subject, the experimenter started timing using a stop-watch. After performing the operation the subject read the answer aloud. If an error was made in a trial, the subject was asked to correct it. Timing stopped when the correct answer was achieved. If required, the experimenter reminded the subject what the 2 numbers were during the trial.

The music playing task required the subject to play a specific song from a set of playlists that were created for this study. Poweramp Music Player app (available on Google Play) [21] was used in this task. The core user interface of Poweramp had four different screens (shown in Fig. 10). The main screen had a menu with options such as Artists, Albums, and Playlists. For this study, only the Playlists option was used. Tapping Playlists displayed the stored playlists. Selecting a playlist led to a screen showing all the songs in that playlist. Clicking on the song title played the song, with the screen showing the media player interface. Some screens had an option of going back to the previous screen, and the media player screen had an option to go back to the main menu.

There were six custom made playlists with distinct names of artists: Bach, Brightman, Jackson, Carpenter, Simon, and Strauss. Each playlist had five to seven songs whose names were preceded by the artist's name. For example, in the Bach playlist, each song name started with the prefix "Bach –". Each trial consisted of playing a requested song, and then changing to a second song. At the start of the trial, the experimenter specified the playlist and the song name to be played. The subject was asked to confirm the name of the playlist and the song after which the trial timing would start. The trial always started at the main menu screen and the subject was instructed to follow the usual sequence of navigation from the main menu to the media player interface screen. After playing the first song successfully, the subject was asked to immediately play another song from a different playlist by going back to the main menu and repeating the same steps. The trial stopped when the subject successfully played the second song and navigated back to the main menu. There were six trials for the music playing task. Similar to the calculation task, the time of the trial and the mistakes made during the trial were recorded. If an error was made, the subject was asked to navigate back and forth as required to correct it so as to successfully complete the trial.

TABLE I
DETAILS OF STUDY SUBJECTS

	Vision Condition	Age	Binocular VA (logMAR)	Right Eye VA (logMAR)	Contrast Sensitivity
1	Normally Sighted	35	0.00	0.00	1.85
2	Normally Sighted	36	0.10	0.10	2.00
3	Normally Sighted	25	-0.12	-0.12	2.08
4	Normally Sighted	34	-0.12	-0.12	1.95
5	Normally Sighted	34	-0.12	0.00	2.15
6	Normally Sighted	25	-0.12	-0.12	2.03
7	Normally Sighted	69	0.00	0.00	1.95
8	Normally Sighted	42	0.20	0.20	2.15
9	Optic nerve atrophy	44	0.82	0.86	1.25
10	Coloboma, glaucoma, nystagmus	50	0.70	0.72	1.78
11	Macular dystrophy, North Carolina dystrophy	39	0.50	0.70	1.83
12	Optic neuropathy	41	0.44	0.48	1.30

For both tasks, the head-motion range in the horizontal and vertical directions, when viewing through the Google Glass were set at 90° and 60°, respectively. Hence, there was no head movement calibration for each subject. However, motion sensor drift occurs with the Google Glass and subjects were instructed to re-center the display as required, and were periodically reminded about this between trials.

The order of the device conditions (smartphone or Google Glass) was counterbalanced for each task. The multiplication operations performed in the calculation task were the same in both conditions. There was little risk of a learning effect confounding the outcome in the other condition because it was difficult to remember the exact numbers. However, to alleviate the risk of any learning effect in the music playing task, we used two different sets of song pairs for the two conditions, and each subject played a pair of songs only once in the study. The two sets were counterbalanced across subjects.

We recruited eight normally sighted individuals and four low-vision patients for this study (subject demographics are given in Table I). All of the low-vision subjects habitually used smartphones and frequently relied upon the smartphone screen magnification accessibility features. The study was approved by the Human Subjects Committee of Massachusetts Eye and Ear and written informed consent was obtained before participation. Each subject was given some training before the experiment for scrolling on a magnified smartphone screen as well as using the Google Glass app for the two tasks. When viewing with the Glass, the letter sizes in the calculation and music playing task were 20/632 and 20/252, respectively. The letter sizes were well above the measured VAs of the subjects. Obtaining subject consent, task instruction and training took about 60 min, while the experimental tasks took an average of 15 min: 5 min for the calculation and 10 min for music playing task.

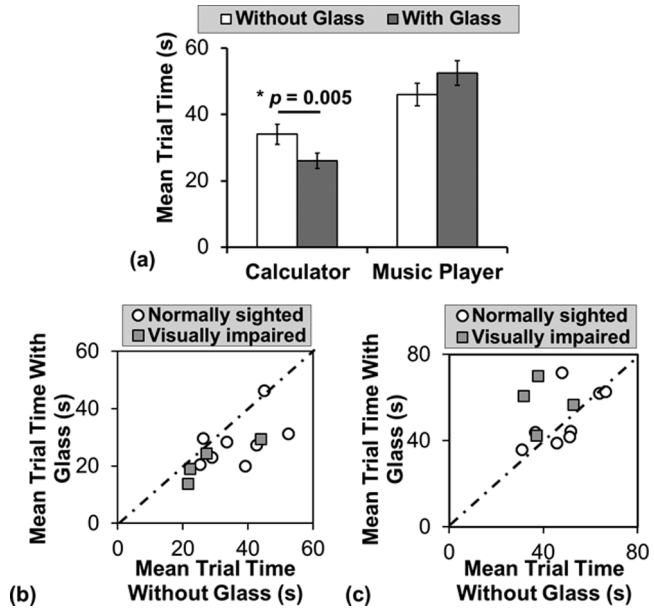

Fig. 11. User evaluation results. (a) Average trial time for the two device conditions compared for the calculation and music playing tasks. The error bars represent the std. error of the mean. (b) Scatter plot of mean trial times with and without Glass for the calculation task, showing shorter trial time with Glass for the majority of the subjects including all the visually impaired participants (points below the dotted line). (c) Scatter plot of mean trial times with and without Glass for the music playing task shows that the trial times were not different for the majority of the subjects (points close to the dotted line). Two visually impaired and one normally sighted subject were substantially slower with Glass on this task.

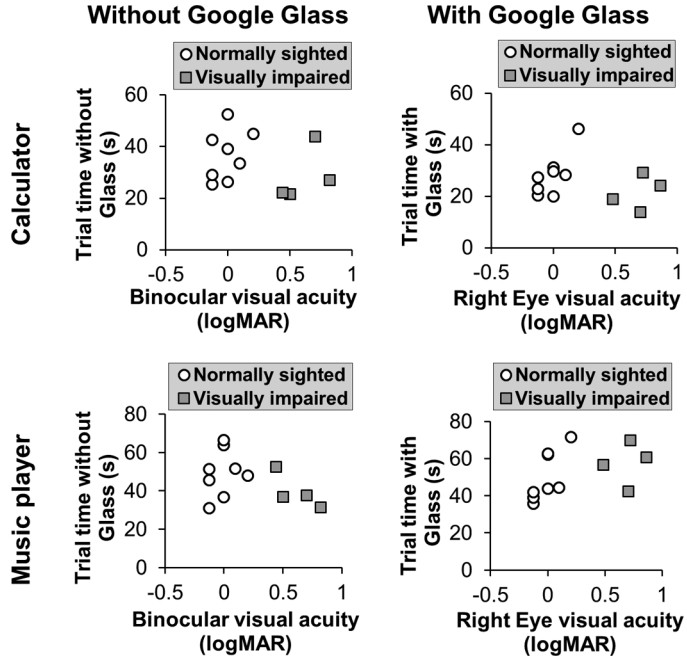

Fig. 12. Relationship between trial time and visual acuity (VA) for calculation and music playing tasks without and with Google Glass. A larger value on the horizontal axis corresponds to worse visual acuity. Range of trial times for the normally sighted and visually impaired subjects overlapped considerably in all cases, and some visually impaired subjects out-performed normally sighted subjects. Thus, subjects' VA did not affect the trial time in our experiments.

B. Results

The mean trial time (in seconds) for all the subjects (normally sighted and visually impaired combined) in the two device conditions: with smartphone directly and with Google Glass, were compared within subjects. Paired t-test was used to determine the differences in the mean trial time between the two experimental conditions.

Fig. 11 shows the user evaluation results for both tasks. For the calculation task, the average trial time with Glass was significantly shorter than without Glass ($\text{mean} \pm \text{standard error}$: without Glass $= 34.0 \pm 3.0$; with Glass $= 26.1 \pm 2.4$; $t(11) = 3.504$, $p = 0.005$). There was no significant difference in the mean trial times between the two conditions in the music playing task [$\text{mean} \pm \text{standard error}$: without Glass $= 46.1 \pm 3.4$; with Glass $= 52.5 \pm 3.7$; $t(11) = -1.559$, $p = 0.147$; Fig. 11(a)]. Fig. 11(b) and (c) shows the comparison between the two conditions for each subject individually with the dotted line representing the $y = x$ line of equal performance in each condition. In the calculation task [Fig. 11(b)], a majority of the points lie below this line indicating shorter trial time with Google Glass. There were only two subjects who recorded slightly higher trial times with Google Glass. In the music playing task [Fig. 11(c)] a majority of the points lie close to the $y = x$ line, indicating that trial times between the two conditions were similar for most subjects. Three subjects (one normally sighted and two visually impaired) performed worse with the Glass than with the smartphone.

Fig. 12 shows the relationship between subjects' VA and average trial time for each task. For the Google Glass condition the right eye VA is considered, as the display is only viewed by the right eye. Based on VA there are two distinct groups: normally sighted subjects' VA ranged from -0.13 to 0.2 logMAR, whereas visually impaired subjects had worse VA between 0.44 and 0.86 logMAR. Subjects with worse VA did not consistently have worse performance than those with better VA. In fact, a trend in the music playing task without Glass, where visually impaired subjects with worse VA were relatively faster may be seen. Overall, visually impaired subjects' performance was within the range of normally sighted subjects' performance for both tasks, and sometimes it was even better than the normally sighted subjects (lower trial time).

VI. Discussion

The impetus for the development of the head controlled screen navigation system was the assumption that intuitive head position feedback may help users navigate magnified displays. The prototype system allows the use of a natural proprioceptive feedback—sense of head position to guide panning. This means that if the user knows approximately where things are on the whole screen, he or she can navigate with ease, even if only a small portion of the screen is visible at any time. For example, in the calculator app, the button for '0' is on the bottom left. The user knows that he or she will be able to find it by directly turning the head approximately toward the lower left corner of the virtually enlarged screen. It is possible for the Glass users to keep themselves properly orientated within the virtually enlarged screen based on proprioceptive feedback from the head, instead of relying on the limited local context on the screen. Of course, knowing the layout of the screen may also help in

scrolling in the right direction on the smartphone, but spatial awareness may not help the hand scrolling gesture as it is done in phone screen coordinates and not in the coordinates of the virtually enlarged screen. In screen magnification software for desktop computers, a small overview map in the corner showing the location of the currently zoomed-in area relative to the whole screen is a commonly used feature to help orient visually impaired users. This feature is not suitable for the phone screen since it is already very small.

Our approach, based on the dimensions of the Glass, is targeted toward people with moderate vision loss, who habitually prefer smartphone magnification over speech based accessibility features. In this context, we used the built-in smartphone magnification accessibility feature as a baseline comparison because the stimuli in both conditions (with and without Glass) are visual and are well matched (the content and magnification is the same in both conditions). The primary difference is in the way interaction with the visual input occurs in each condition: using touch gestures on a smartphone screen versus head motion (proprioceptive feedback) with the Google Glass. Subjects were significantly faster using the Glass in the calculation task, but there was no significant difference in trial time in the music playing task. The layout of the calculator app was familiar, easier to remember, and remained fixed throughout the experiment. The music player, on the other hand, had multiple screen layouts that changed based on the options selected by the users. The users had to search for the playlist from the menu and a particular song within the selected playlist. Unlike the calculator app, it was almost impossible for participants to acquire knowledge of the layout of the music player app during the short study. The different results for the calculator and music player tasks suggest that knowledge of screen layout could be an important factor affecting performance with the head controlled screen zoom method. We speculate that when users become familiar with the order (layout) of playlists and songs, their performance with the Glass screen zoom app might be improved.

Two other factors could have affected the performance of the visually impaired subjects. First, the visually impaired subjects were used to working with the built-in screen zoom since they used smartphone magnification in their daily activities. As a group, their average trial time with the smartphone was lower than normally sighted subjects for both tasks (Fig. 12). Second, the lower contrast offered by the Google Glass see-through display could have affected subject performance, especially in the music player task where reading was involved. Based on these two factors, it could be argued that the study could have been biased in favor of manual scrolling on the smartphone, and that the beneficial effect we found with head motion based navigation for the calculation task is more meaningful in this context. It can be further improved if visually impaired users become more accustomed with the Google Glass, and the Glass display contrast is increased (e.g., by increasing the brightness or making it opaque).

Our user evaluation pilot study was limited by the small sample size and the variety of tasks that were tested. As there were only four visually impaired subjects, the small sample size is not suitable for statistical analyses. However, based on trial time with or without the Glass, there was no distinction between the two subject groups [Fig. 11(b) and (c)]. Some visually impaired subjects recorded faster trial times than some normally sighted subjects, suggesting that magnification indeed compensated for the acuity difference between the two groups. While there might be a trend of less benefit of the Glass with lower visual acuity for the music player, more data are needed to validate this trend.

The current prototype implementation of the app has some limitations. First, the commonly used swiping gesture control has not been implemented on the Glass. If scrolling is needed, e.g., to scroll on a webpage, users will need to swipe on phone. However, sending swiping control to smartphones is technically feasible as the Glass supports horizontal and vertical swiping gestures. Second, the relatively low frame rate of our prototype, primarily due to the screenshot capture process, may limit the current implementation to tasks that do not involve highly dynamic screen content. For example, videos would not work well with the current prototype. Technically, the frame rate can be largely improved through engineering development. Nowadays remote desktop control techniques can provide reasonably fast frame rates. Furthermore, our current method of screenshot capture (from the display buffer) requires root access to the phone, which may not be possible for some devices and may not be desirable for some users.

The purpose of this study was to evaluate the potential value and identify the limitations of a head controlled navigation method for magnified screens. While Google Glass explorer edition was used in this work, the concept will persist irrespective of the underlying hardware. The implementation is essentially an Android app that makes it portable to other compliant hardware. Interest in smart augmented reality/virtual reality glasses has increased in recent years and we anticipate the technique described in this paper can be implemented in other newer models that will be introduced in the market by a variety of companies, including a newer version of Google Glass [22]. We envision that with the availability of better and more compliant hardware and software platforms, a widely available and practically useful system that overcomes the above mentioned limitations can be implemented for visually impaired users when dealing with their magnified smartphone displays.

VII. Conclusion

With the increasing use of smartphones among people with low-vision, there is a need to address the limitations of conventional screen zoom accessibility features: loss of context and slow navigation time. We have implemented an app to project magnified smartphone screens to Google Glass, with which the users can move their head in the space to view the corresponding portion of the magnified mobile screen. We argue that proprioceptive feedback can be useful in zoom-panning applications, and it can be effectively harnessed via advances in wearable computing technology. Our evaluation study with 12 subjects showed that for the same level of magnification, the head-motion based navigation method reduced the average trial time compared to conventional manual scrolling for the calculation task (by about 28%), but not for the music playing task. One of

the possible reasons could be that the screen layout of the calculator was known beforehand and straightforward to remember resulting in reduced trial time with head-motion based screen navigation. Further evaluation involving a variety of tasks is necessary in order to fully understand the benefit of proprioceptive feedback in screen navigation. Future work includes implementing more gestures on the Google Glass to interact with smartphones and comparing the effectiveness of head-motion based navigation with other commonly used voice-based mobile accessibility features.

REFERENCES

[1] S. Smallfield, K. Clem, and A. Myers, "Occupational therapy interventions to improve the reading ability of older adults with low vision: A systematic review," *Am. J. Occupat. Ther.*, vol. 67, pp. 288–295, 2013.

[2] N. X. Nguyen, M. Weismann, and S. Trauzettel-Klosinski, "Improvement of reading speed after providing of low vision aids in patients with age-related macular degeneration," *Acta Ophthalmologica*, vol. 87, pp. 849–853, 2009.

[3] G. L. Goodrich and J. Kirby, "A comparison of patient reading performance and preference: optical devices, handheld CCTV (Innoventions Magni-Cam), or stand-mounted CCTV (Optelec Clearview or TSI Genie)," *Optometry*, vol. 72, pp. 519–28, 2001.

[4] P. Blenkhorn, D. G. Evans, and A. Baude, "Full-screen magnification for windows using DirectX overlays," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 10, no. 4, pp. 225–231, Dec. 2002.

[5] P. Blenkhorn and D. G. Evans, "A screen magnifier using "High Level" implementation techniques," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 14, no. 4, pp. 501–504, Dec. 2006.

[6] ZoomText Aisquared [Online]. Available: http://www.aisquared.com/zoomtext

[7] SuperNova Dolphin [Online]. Available: http://www.yourdolphin.com/ productdetailnew.asp?id=3

[8] MAGic Freedom Scientific [Online]. Available: http://www.freedom scientific.com/Products/LowVision/MAGic

[9] Accessibility Apple, Inc. [Online]. Available: https://www.apple.com /accessibility/osx/

[10] Windows Magnifier Microsoft [Online]. Available: http://windows.microsoft. com/en-us/windows/make-screen-items-bigger-magnifier#1TC=windows-7

[11] J. Morris, J. Mueller, M. L. Jones, and B. Lippincott, "Wireless technology use and disability: Results from a national survey," in *J. Technol. Persons Disabilities, Annu. Int. Technol. Persons Disabilities Conf.*, I. Barnard, Ed. *et al.*, 2013, pp. 70–80.

[12] M. D. Crossland, R. S. Silva, and A. F. Macedo, "Smartphone, tablet computer and e-reader use by people with vision impairment," *Ophthalmic Physiol. Opt.*, vol. 34, pp. 552–557, 2014.

[13] V. Braimah, J. Robinson, R. Chun, and W. M. Jay, "Usage of accessibility options for the iPhone/iPad in a visually impaired population," *Assoc. Res. Vis. Ophtalmol.*, 2014.

[14] P. Baudisch, N. Good, V. Bellotti, and P. Schraedley, "Keeping things in context: A comparative evaluation of focus plus context screens, overviews, and zooming," in *Proc. ACM SIGCHI Conf. Human Factors Comput. Syst.*, 2002, pp. 259–266.

[15] R. Ball, C. North, and D. A. Bowman, "Move to improve: Promoting physical navigation to increase user performance with large displays," in *Proc. ACM SIGCHI Conf. Human Factors Comput. Syst.*, 2007, pp. 191–200.

[16] D. Raja, D. A. Bowman, J. Lucas, and C. North, "Exploring the benefits of immersion in abstract information visualization," in *Proc. Immersive Projection Technol. Workshop*, 2004.

[17] Google Glass Goole, Inc. [Online]. Available: https://www.google.com/ glass/start/

[18] M. Miyakawa, Y. Maeda, Y. Miyazawa, and J. Hori, "A smart video magnifier controlled by the visibility signal of a low vision user," in *Proc. 28th IEEE EMBS Annu. Int. Conf.*, 2006, pp. 4213–4216.

[19] D. Fono and R. Vertegaal, "EyeWindows: Evaluation of eye-controlled zooming windows for focus selection," in *Proc. ACM SIGCHI Conf. Human Factors Comput. Syst.*, 2005, pp. 151–160.

[20] *Calculator Plus*, [Online]. Available: https://play.google.com/store/apps/details?id=com.digitalchemy.calculator.freedecimal&hl=en

[21] *Poweramp Music Player*, [Online]. Available: https://play.google.com/store/apps/details?id=com.maxmpz.audioplayer&hl=en

[22] L. Eadicicco, See the New Version of Google's Wildest Product 2015 [Online]. Available: http://time.com/4163067/google-glass-2-photos-2015/

Shrinivas Pundlik received the B.E. degree in electronics from University of Pune, Pune, India, in 2002, and the M.S. and Ph.D. degrees in electrical engineering from Clemson University, Clemson, SC, USA, in 2005 and 2009, respectively.

He is currently working as a post-doctoral fellow of Harvard Medical School at the Schepens Eye Research Institute, Boston, MA, USA. His current research focuses on computer vision, vision science, and vision rehabilitation.

HuaQi Yi received the B.S. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2012, and the M.S. degree in computer science from Northeastern University, Boston, MA, USA, in 2014.

After graduation, he has been focusing on mobile app development.

Rui Liu received the M.D. and the Ph.D. degrees in ophthalmology from Fudan University, Shanghai, China, in 2008.

He was a post-doctoral fellow of Harvard Medical School when the presented work was conducted. He is currently an ophthalmologist at Eye and ENT Hospital of Fudan University, Shanghai, China. His research interests include vision science, vision rehabilitation and mechanisms of eye diseases such as myopia, strabismus and amblyopia.

Eli Peli received the M.S. degree in electrical engineering from the Technion-Israel Institute of Technology, Haifa, Israel, in 1979, and the OD degree from New England College of Optometry, Boston, MA, USA, in 1983.

He is the Moakley Scholar in Aging Eye Research, and Professor of Ophthalmology at Harvard Medical School. Since 1983, he has been caring for visually impaired patients as the Director of the Vision Rehabilitation Service at Tufts Medical Center Hospitals, Boston, MA, USA. His principal research interests are image processing in relation to visual function and clinical psychophysics in low-vision rehabilitation, image understanding and evaluation of display–vision interaction.

Gang Luo received the Ph.D. degree from Chongqing University, Chongqing, China, in 1997.

He is an Associate Professor at The Schepens Eye Research Institute, Harvard Medical School, Boston, MA, USA. His primary research interests include vision science and vision assistive technology, and vision care technology based on mobile platform.