Smart Phone Interaction with Registered Displays

This article describes the theory and practice of display registration with smart phones, findings in initial user studies, and opportunities for developing markerless display registration schemes.

Nick Pears
University of York

Daniel G. Jackson and Patrick Olivier
Newcastle University

Imagine standing in front of a real estate agent's shop window one evening. Looking at the computer driven display in the window, you see several interesting properties. You want to download the related information and leave your details so the staff can contact you to arrange viewings. You also want to scroll through the other properties, using all the features of the large public display. Of course, you want to interact with the display directly, and seamlessly move data between it and your own smart phone—and even have your phone become part of the real estate agent's display. Composing the two displays would circumvent the need to learn new mappings and interaction techniques and, in essence, you could directly interact with this public display through your smart phone's screen. Indeed, by observing users performing tasks on other classes of situated displays, such as digital tabletops, researchers have noted the fluid mix of activities and rapid switching between tasks.[1] Composing the displays means not having to switch between the smart phone and the public display, and it leverages our familiarity with smart phone input techniques (such as button configurations and stylus).

In fact, the technologies necessary to realize the direct interaction in our real estate agent example are standard components of all smart phones: a rear-mounted camera and wireless connectivity. If we point the smart phone camera at a public display, we can treat the public display and the smart phone image as the same visual element, providing that we can communicate data and interaction events between the phone and the display, and quickly and reliably compute the geometric mapping between the smart phone's image and the public display. Finding this geometric mapping is an image registration problem that we call *display registration*. If we can register the two displays, we can open the door to a host of potential interaction techniques, from the devices' direct manipulation metaphors to the smart phone's use as a physical tool or 6-degree-of-freedom (DOF) flying mouse. This assumes a trusted and secure wireless communication between the smart phone and public display. In this respect, a related and potentially problematic issue is the usability of Bluetooth configuration and authentication. However, recent work on spontaneous security is beginning to address these issues.[2] (For other work in bridging the gap between personal devices and situated displays, see the "Related Work in Mobile Interaction" sidebar.)

Display Registration

Figure 1 illustrates display registration technology, whereby we can treat the smart phone screen as an almost indistinguishable part of another system's display (that is, the public dis-

 Published by the IEEE CS ■ 1536-1268/09/$25.00

Related Work in Smart Phone Interaction

Much work in pervasive computing has attempted to bridge the gap between personal devices and situated displays. Early work tended to augment handhelds' limited capabilities with enhanced sensing or communication capabilities. For example, George Fitzmaurice's pioneering work on situated information spaces used a 6-degree-of-freedom (DOF) positional sensing device to enable interaction with a virtual 3D workspace.[1] Ken Hinckley and his colleagues used infrared proximity sensors, touch sensors, and tilt sensors to initiate and control smart phone interaction.[2] Ka-Ping Yee proposed using pen interactions and peephole displays on large virtual workspaces.[3] And Giorgio De Michelis and his colleagues used infrared-equipped phones in interactions with large displays, where the phone served as a general-purpose remote control, sending commands to the display.[4]

Many recent proposals for integrating phones and situated displays have sought to use visually-controlled interaction, such as Yuichi Yoshida and his colleagues' "mobile magic hand,"[5] which lets users manipulate virtual objects using the measured optical flow in an image sequence when observing a visual code. Jingtao Wang and his colleagues developed motion-estimation techniques for a camera-equipped phone that derive from full-search block matching, similar to those MPEG video encoders use.[6]And, finally, Michael Rohs developed a system in which the phone displays a marker on its screen and an external camera (attached to the public display) tracks the position of the marker.[7]

REFERENCES

1. G.W. Fitzmaurice, "Situated Information Spaces and Spatially Aware Palmtop Computers," *Comm. ACM*, 1993, vol. 36, no. 7, pp. 39–49.
2. K. Hinckley et al., "Sensing Techniques for Mobile Interaction," *Proc. 13th Ann. ACM Symp. User Interface Software and Technology*, ACM Press, 2000, pp. 91–100.
3. K. Yee, "Peephole Displays: Pen Interaction on Spatially Aware Handheld Computers," *SIGCHI Conf. Human Factors in Computing Systems*, ACM Press, 2003, pp. 1–8.
4. G. DeMichelis, M. Loregian, and P. Martini, "Directional Interaction with Large Displays using Mobile Phones," *Proc. 4th IEEE Int'l Conf. Pervasive Computing and Comm. Workshops* (PERCOMW 06), IEEE CS Press, 2006, pp. 196–200.
5. Y. Yoshida, K. Miyaoku, and T. Satou, "Mobile Magic Hand: Camera Phone Based Interaction using Visual Code and Optical Flow," *Human-Computer Interaction Part II*, LNCS 4551, 2007, pp. 513–521.
6. J. Wang, S. Zhai, and J. Canny, "Camera Phone Based Motion Sensing: Interaction Techniques, Applications and Performance Study," *Proc. 19th Ann. ACM Symp. User Interface Software and Technology* (UIST 06), ACM Press, 2006, pp. 101–110.
7. M. Rohs, "Linking Physical and Virtual Worlds with Visual Markers and Handheld Devices," doctoral dissertation, Dept. of Computer Science, ETH Zurich, 2005.

play). By pointing the smart phone rear-mounted camera at the public display screen, we can manipulate elements of the resulting image as if it were part of the smart phone display. The registration of the two displays means that for any position (pixel) on the public display, we know its corresponding position in the smart phone's image of that display, and vice-versa. We can then invoke movements of the smart phone, and button and stylus actions, in developing direct, natural interaction schemes both within and between the two devices. Examples of such interaction schemes include:

- **Direct touch.** The very essence of display registration is that we can achieve direct interaction with interface elements on the public display by performing these actions on the smart phone's image. Direct touch on the smart phone's image of the public display is indistinguishable from directly interacting with the public display itself.
- **Data exchange.** We can divide the smart phone's screen into two regions: one representing the smart phone itself (such as for showing folder and file icons) and one containing the image of the public display. We can directly manipulate files and folders within and between the two regions.
- **Tangible tools.** The display registration lets us compute a 6-DOF position and orientation (pose) of the smart phone, with which we can mediate interactions using the smart phone as a 2D or 3D mouse, or some other tangible tool. For example, we can select images on the public display and we can rotate and scale them by physically manipulating the phone (for instance, while selecting an image, rotating the phone, and moving it forward and backward).

Achieving Display Registration

To describe how we achieve display registration, we consider the common case of a planar smart phone image sensor and a planar public display. If we can consider the imaging process associated with the camera lens to be idealized, we can apply the ideal "pinhole" camera model. This models the path of the light traveling from the display to the smart phone image sensor as a straight line passing through the center of the smart phone lens. (This model is

(a) (b)

Figure 1. Images being moved, rotated and scaled using display registration.

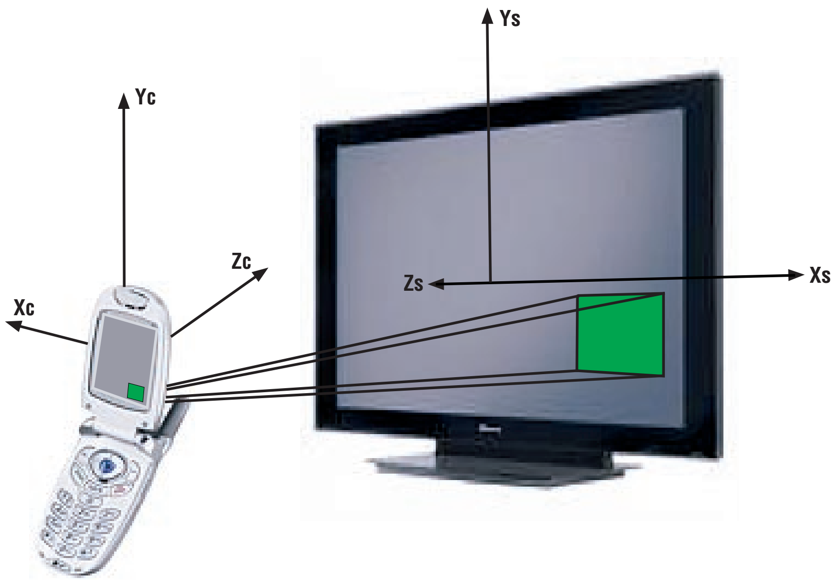


Figure 2. The geometry of a smart phone viewing a marker on a public display.

accurate because the smart phone imaging machinery compensates for the smart phone lens distortion.) In a sense, this lets us think of the center of each pixel in the smart phone image as pointing to some point on the public display by projecting a straight line from that pixel through the optical center of the lens (see Figure 2). Clearly, we need to find the (invertible) plane-to-plane projective mapping that lets us compute the display registration. This projective mapping, called a *planar projectivity* or *planar homography*, depends on two sets of parameters:

- *Extrinsic properties*. The part of the public display imaged on the smart phone depends on what the smart phone is looking at. More formally, this is the 6-DOF pose of the public display relative to a Euclidean frame attached to the smart phone camera's optical center. This pose consists of three Euclidean translation parameters and three orientation parameters (*pan*, *tilt*, and *roll*).
- *Intrinsic properties*. The direction of light coming from a particular pixel toward the smart phone camera depends on that pixel's position in the display's Euclidean frame. When that light is imaged, the smart phone image pixel that the light falls on depends on the focal length of the smart phone lens, the position of the image sensor relative to the camera optical axis, and the spacing of the smart phone image pixels in both dimensions of the imaging plane.

Figure 2 illustrates this process and defines two Euclidean 3D frames, in which 3D points are described in metric units. Also, in this figure, we have defined a "virtual image plane" for the camera, defined in front of the camera lens, so we can imagine a virtual image with the same orientation as the public display. Consider a point on the public display relative to a coordinate frame attached to the plane of the display; the description of this point in the smart phone camera frame is simply a Euclidean coordinate transformation:

$$\mathbf{X}_c = \mathbf{E}\mathbf{X}_s \quad (1)$$

where

$$E = \begin{pmatrix} r_1 & r_2 & r_3 & -RT \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2)$$

$\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ are the 3 × 1 column vectors of the rotation matrix, **R**. **R** aligns the camera frame to the screen frame, and **–RT** is the 3 × 1 translation vector, computed from the rotation matrix and the translation vector **T**, describing the camera origin's position in the screen frame. $\mathbf{X}_c = [X_c, Y_c, Z_c, 1]^T$ is the position of the screen point in the camera frame, and $\mathbf{X}_s$ is a similarly defined 4 × 1 vector. The expression of these 3D positions in 4D homogenous coordinates allows, among other things, the packaging of a 3 × 3 rotation matrix and 3 × 1 translation vector into a single linear transform, represented by the 4 × 4 matrix, **E**. We see from Figure 2 that $Z_s = 0$, so we can delete the third column of **E** to give a 4 × 3 matrix and describe a metric position on the public display as $\mathbf{X}_s = [X_s, Y_s, 1]^T$.

The public display metric position, $\mathbf{X}_s$, derives from the screen pixel coordinates, $\mathbf{x}_s = [x_s, y_s, 1]^T$ using an affine transformation—namely,

$$\mathbf{X}_s = \mathbf{K}_s\mathbf{x}_s \quad (3)$$

where, assuming no shear:

$$\mathbf{K_s} = \begin{pmatrix} \alpha_x & 0 & X_0 \\ 0 & \alpha_y & Y_0 \\ 0 & 0 & 1 \end{pmatrix} \quad (4)$$

Here, a_x and a_y are the number of metric units per public display pixel in the X and Y dimensions respectively, and (X_0, Y_0) are the metric coordinates of the public display's origin. The screen point, described in the metric camera frame, $\mathbf{X}_c$, can be projected down into the pixel coordinates within the image $\mathbf{x}_c$, using a projection matrix, $\mathbf{K}_c$, which contains the camera intrinsic parameters:

$$l\mathbf{x}_c = \mathbf{K}_c\mathbf{X}_c \quad (5)$$

where, assuming no shear, the affine transformation, $\mathbf{K}_c$, is given as

$$\mathrm{K_c} = \begin{pmatrix} \beta_x f & 0 & 0 & u_0 \\ 0 & \beta_y f & 0 & v_0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad (6)$$

Here, f is the focal length of the smart phone lens. b_x and b_y are the number of smart phone pixels per metric length in the x and y directions of the camera frame, respectively, and (u_0, v_0) are the pixel coordinates of where the optical axis intersects the image plane. $\mathbf{x}_c = [u, v, 1]^T$ are the homogenous coordinates of the image position (in pixel units)—so, $l = Z_c$, the depth of the point in the camera frame. If we concatenate the effects of the extrinsic parameters (pose) and intrinsic parameters (camera/display), we have

$$l\mathbf{x}_c = \mathbf{K}_c\mathbf{E}\mathbf{K}_s\mathbf{x}_s = \mathbf{H}\mathbf{x}_s \quad (7)$$

Thus, we can map a point on the public display, expressed in pixels ($\mathbf{x}_s$), to a camera pixel position ($\mathbf{x}_c$) by a 3 × 3 matrix **H**, which simultaneously encodes the relative pose between the smart phone and public display and the intrinsic parameters of both the smart phone and public display. A crucial point is that, to compute any instance of such a mapping, we don't need the intrinsic and extrinsic parameters explicitly, only their combined values in the homography matrix **H**. Now, in projective geometry, a homogenous point $\mathbf{x}_c$ and some arbitrary scaling of that point $\lambda\mathbf{x}_c$ are equivalent. This implies that we can't disambiguate between a small public display that's close to the smart phone and one which is **n**-times as large and **n**-times the distance from the smart phone.

This means that the 3 × 3 matrix **H** is defined up to a scale factor and so has eight degrees of freedom rather than nine. Thus we can estimate the matrix using standard linear methods if we know four corresponding points between the smart phone image and public display, with the constraint that no three are collinear. In this case, we have eight independent constraints, and **H** is fully defined (up to scale). Additional corresponding points can yield a more accurate estimate of **H** using some variant of a least-squares technique. Richard Hartley and Andrew Zisserman detail various estimation techniques for **H**.[3] Typically, these involve rearranging the equation $l\mathbf{x}_c = \mathbf{H}\mathbf{x}_s$ into the form $\mathbf{A}\mathbf{h} = \mathbf{0}$, where **h** is a 9 × 1 vector of the elements of **H**, and **A** is a matrix generated from n pixel-based correspondences ($\mathbf{x}_s$, $\mathbf{x}_c$). We can then use singular value decomposition to solve for **h**. Obviously, we can determine a solution only up to a nonzero scale factor and, typically, select a scale by a requirement on the norm of the values in **h**—namely, the sum of the square of their values is equal to one.

The main idea is to make the game persistently available to the players anytime. ALIGN THE TOP BAR WITH BASELINE.

Marker-based Display Registration

In *marker-based display registration*, the public display must maintain a dynamic display of four distinctively colored reference targets, no three of which are collinear, which the smart phone can easily detect, segment, and localize in its image. We can transmit

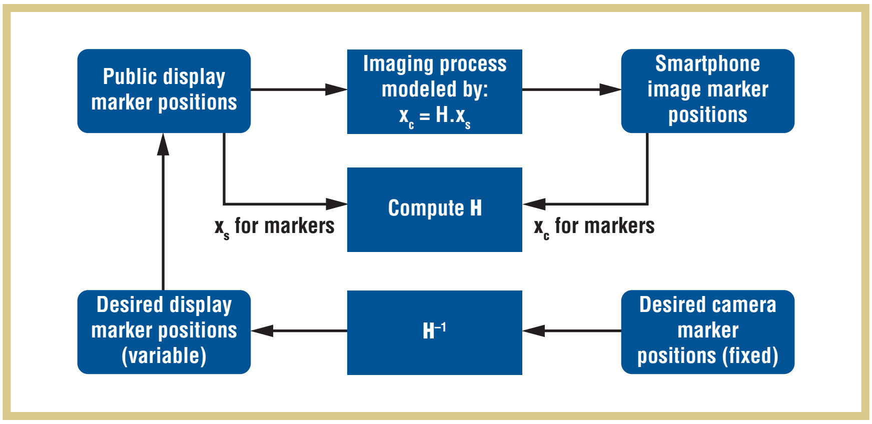


Figure 3. The process that updates the public display marker positions, so that they appear at a fixed position and size in the smart phone image.

these marker positions to the public display, which knows where the targets were originally displayed, if we use some temporal synchronization scheme (which we describe later). We can then apply a planar homography estimation method to register the smart phone and public display without previously calibrating the smart phone camera.

An important point is that, because we know the homography transformation between the public display and smart phone image, we can change the markers on the public display, such that the markers' shape, size, and position is constant in the smart phone image irrespective of any changes in camera viewing pose and camera zoom. To implement this, once we have computed the homography, **H**, that maps public display pixels to smart phone pixels, we also can compute the inverse mapping, $\mathbf{H}^{-1}$, which maps smart phone pixels to public display pixels. We can then map the desired positions of the markers in the smart phone image, through the inverse homography, to public display positions. Figure 3 illustrates this process. The end result is that the smart phone markers vary only slightly, whereas the public display markers vary significantly. This process leads to more reliable detection of the markers, because, for example, they don't become too small to detect as the smart phone moves away from the public display. In this case, the markers on the public display automatically adapt by increasing in size to keep the imaged marker size constant.

A Prototype System

We have implemented several marker-based prototype systems that support a single user, one of which is shown in Figure 1. For the markers, we chose four squares of a distinctive green color, each of which can be either hollow (H) or filled (F). We allow individual squares to have two visual states (hollow and filled) for two reasons:

- **Unambiguous target orientation.** A pattern of four squares, arranged in a square, has a four-fold rotational ambiguity. We need to break the rotational symmetry of this pattern to uniquely identify the orientation (roll) of the smart phone relative to the public display. Given that we have four squares, each of which has two states (hollow/filled), there are 16 patterns, 12 of which have no rotational ambiguity (but we only use three of these, for reasons we describe below).
- **Temporal synchronization.** Because the markers move around the public display, we need a temporal synchronization mechanism so that the system knows the correspondence between the target displayed on the public display and that detected on the smart phone. Only with the correct correspondences can we maximize the stability and dynamic response of the tracking. To do this, the public display shows a certain pattern, say FFFH, corresponding to three filled and one hollow square, and it won't change this arrangement of "H" and "F" until the smart phone has successfully detected and reported the position of this pattern over Bluetooth. In fact, we cycle between three of the 12 rotationally unambiguous patterns: FFFH, FFHH, and FHHH. We deliberately chose these patterns to have different numbers of "F" and "H" in their content to avoid interpreting them as rotations of each other.

System Operation

The following is the sequence of events for the system operation:

- The smart phone and public display establish a communication channel over a Bluetooth link.
- The initialization process starts with the public display systematically moving the target around its screen, starting from the center and working out toward the edges. The user starts by aiming the smart phone toward the center of the public display.
- When the smart phone acquires the target, it transmits the 2D image positions of the four centers of the target squares (eight values) in the target to the public display. The public display then associates the corresponding four publically displayed positions with these smart phone target positions and computes the plane-to-plane homography mapping between them.
- The public display computes the public display positions corresponding to the desired corner positions of the markers in the smart phone image. This indicates how the target

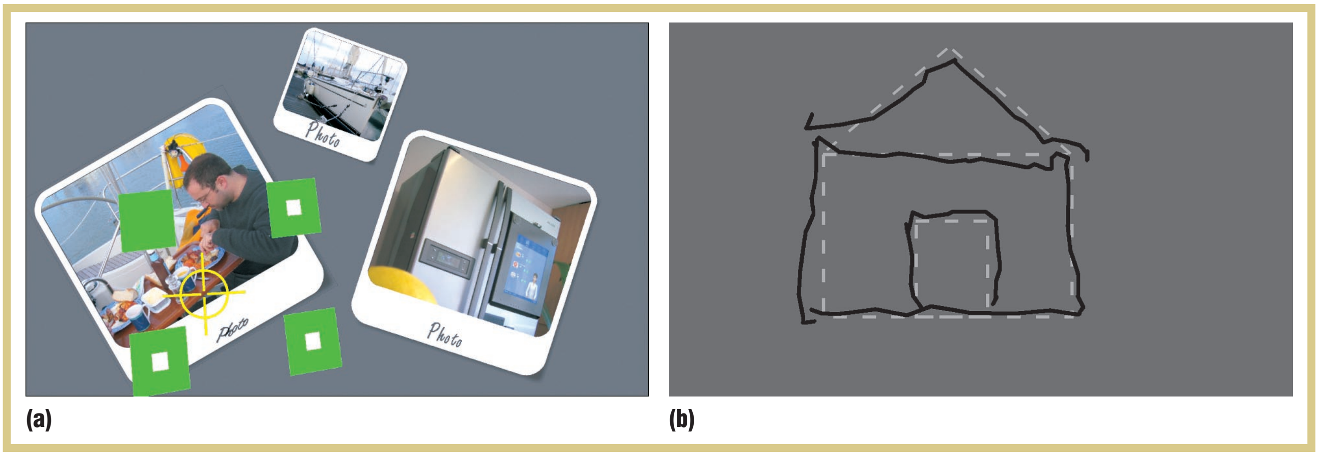


Figure 4. User study tasks: (a) a photo montage, where photos must be moved, rotated and scaled to a target configuration and (b) a drawing task, where a target outline of a house (white dashed line) must be traced with the smart phone (black solid line).

pattern should appear in the public display for the pattern to remain constant in shape and size on the smart phone image.

- For further operation cycles, the four displayed target centers switch between different hollow/filled patterns so we can determine the correct correspondence between the displayed target and detected target.

In our prototype, the system worked from zero distance (the smart phone could be placed against the display surface) and was usable up to a distance of 4m from a 48-inch (1280 x 960) plasma display.

Target Segmentation and Acquisition

We modeled the target color detected on the smart phone using RG-chromaticity color space. In this space, the red and green color components are normalized by dividing by intensity, which is the sum of the RGB components. This gives some immunity to intensity variations. In our approach, we divide the RG-plane in color space into bins and manually select the image of the targets. All of the manually selected pixels populate these bins to give a color model as a histogram in RG-space. We can thus determine whether a pixel falls within the modeled color space and classify it as belonging to the target or not. The simple approach that we use is to find the mean pixel position for the color-segmented pixels and divide the image into four (not necessarily equal) segments in directions associated with the target's tracked orientation. The mean positions in these segmented regions correspond to the centers of the four square markers, which is the information we need to compute the homography.

User Studies

We performed an initial usability evaluation using a think-aloud protocol, in which participants described their observations, thoughts, and actions as they interacted with our first implementation. We asked four participants to each complete two tasks on a PC with a 17" LCD display communicating with the smart phone over Bluetooth. The first task involved a photo montage editing task, in which three digital photographs were laid out in a particular starting position, and participants rearranged the photos to resemble a target configuration with different positions, orientations, and scale (see Figure 4). The participants altered the images by translating and rotating the phone parallel to the display and by translating the phone toward or away from the display (that is, they used the smart phone as a 4-DOF mouse). In the second task, they used the system to replicate a specific outline drawing of a house in Microsoft Paint. Participants were required only to make their own brush strokes using the smart phone (see Figure 4).

All four participants appeared to comprehend the system's basic principle after a brief demonstration, and three completed both tasks satisfactorily on the first attempt. The main feedback surrounded the markers' aesthetics and the difficulty of scaling the image using forward and backward movements. In this initial session, we observed some registration errors, mostly resulting from the participant moving the smart phone faster than the system could update the markers. This first implementation was realized on a Siemens SX1 smart phone, with a series 60 phone processor (130 MHz TI OMAP 310), running Symbian OS V6.1. The 30° field of view and frame rate of 8 Hz allowed only lateral motion of roughly 0.3 m/s at the 0.5 m distance from which the participants interacted with the 17" display in the study.

Our second implementation and study used a touch-screen Windows Mobile device with built-in camera (528 MHz Qualcomm processor), from a distance of between 1m to

2m from a large 48" plasma display, which is a realistic public display scenario. In the second study, we sought to evaluate the participants' ability to understand and use the device in a "walk up and use" scenario. We briefly described the system to 10 participants (20- to 45-year-old regular mobile-phone users) and asked them to perform the two tasks from the first study. We modified the interaction technique slightly, allowing users to use the device's stylus, and in the photo montage application the images were dragged from a region of the smart phone display onto the public display (see Figure 1).

All participants successfully completed both tasks (photo montage completion time: 70 ± 20.4s; drawing time: 25 ± 8.3s). With this improved implementation and higher performance hardware, only four registration errors occurred in which the system lost track of the markers (all during the first task). These errors resulted in the system resetting the markers to their home position, delaying each user by 2 to 3 seconds. Strikingly, the study highlighted the ease in which participants, unfamiliar with the system, were able to comprehend and successfully use the system with just a brief introduction.

We've found that if we move the smart phone so its image plane is approximately parallel with the public display, we can easily make image-based measurements that yield information about the smart phone's motion relative to the public display. This image-based approach gives us four degrees of freedom (three translation and one in-plane rotation), and we've used this effectively in our prototype system. However, if we want out-of-plane rotations (such as pan or tilt), it's difficult to disambiguate these motions from translations without using more information about the imaging process. If we recall that the homography matrix simultaneously encodes both the 6-DOF pose and camera/screen intrinsic parameters, and if we know the camera/screen intrinsic parameters, we can extract the 6-DOF pose from the homography matrix. This is indeed true, and we can find camera parameters from calibrating the camera.[3] Our current prototype doesn't use 6-DOF metric pose extraction, but we plan to implement it in future work.

> The main idea is to make the game persistently available to the players anytime. ALIGN THE TOP BAR WITH BASELINE.

The most challenging part of future development is to create a system that can operate without the use of special markers to enable reliable display registration. This is an important progression: it would remove user distractions and create more opportunities for multiuser interaction. In a markerless system, you could achieve display registration using one of several techniques in the computer-vision literature. Perhaps the simplest approach is to use corner extraction[4] followed by matching across the two views. The obvious difficulty is solving the correspondence problem: which corners in the public display match the corners in the smart phone image? The spatial arrangement of corners could be a matching constraint—for example, five corners in a general position provide a pair of cross-ratio invariants,[5] although cross-ratio computation is noise sensitive. Rather than using the spatial arrangement of corners, we could compute specific features in the image that are distinctive and invariant to the imaging process. Several researchers have formulated invariant features, such as Cordelia Schmid and Roger Mohr's rotationally symmetric Gaussian derivatives[6] and Adam Baumberg's second moment matrix,[7] which gives invariance to affine transforms. David Lowe's scale invariant feature transform is perhaps the most widely-used of these.[8] Its features are invariant to similarity transforms (translation, rotation, and scale changes). Although this technique also provides some robustness to affine transforms, large non-affine distortions (caused by large pan and tilt smart phone rotations) are likely to cause the system to lose track. Determining such performance bounds is part of our future plans. Finally, we envisage that we can arrange the design of the public display's windows, icons, and background so they maximize the feature detection and matching to produce a markerless system that enables direct interaction via an invisible display registration process. ▪

REFERENCES

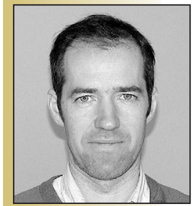
1. A.N. Sulaiman and P. Olivier, "Attribute Gates," *UIST '08: Proc. 21st Ann. ACM Symp. User Interface Software and Technology*, ACM Press, 2008, pp. 57–66.
2. R. Mayrhofer and H. Gellersen, "Shake Well Before Use: Authentication based on Accelerometer Data," *Proc. Pervasive*, LNCS 4480, Springer, 2007, pp. 144–161.
3. R.I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, second ed., 2004, Cambridge University Press.
4. C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Proc. 4th Alvey Vision Conf.*, British Machine Vision Assoc. and Society for Pattern Recognition, 1988, pp. 147–151.
5. D. Sinclair and A. Blake, "Quantitative Planar Region Detection," *Int'l J. Com-*

puter Vision, vol. 18, no. 1, 1996, pp. 77–91.

6. C. Schmid and R. Mohr, “Local Gray-Value Invariants for Image Retrieval,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 5, 1997, pp. 530–535.
7. A. Baumberg, “Reliable Feature Matching across Widely Separated Views,” *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, IEEE Press, 2000, pp. 774–781.
8. D.G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *Int’l J. Computer Vision*, vol. 2, no. 60, 2004, pp. 91–110.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.

the AUTHORS



Nick Pears is a lecturer at the University of York. His research interests include computer vision and pattern recognition. Pears received his PhD in engineering (robotics) from the University of Durham. Contact him at nep@cs.york.ac.uk.

Daniel G. Jackson is a senior researcher at Newcastle University. His research interests include human-computer interaction and pervasive computing. Jackson received his MEng in computer systems and software engineering from the University of York Contact him at d.g.jackson@ncl.ac.uk.

Patrick Olivier is a reader at Newcastle University. His research interests include human-computer interaction, computer graphics, and artificial intelligence. Olivier received his PhD in language engineering from the University of Manchester. Contact him at p.l.olivier@ncl.ac.uk.