MOBILE DEVICE AND INTELLIGENT DISPLAY INTERACTION VIA SCALE-INVARIANT IMAGE FEATURE MATCHING

Leigh Herbert, Nick Pears
Department of Computer Science, University of York, Deramore Lane, York, UK
leigh.herbert@baml.com, nep@cs.york.ac.uk

Dan Jackson, Patrick Olivier
School of Computing Science, Culture Lab, Newcastle University, King's Walk, Newcastle Upon Tyne, UK
d.g.jackson@newcastle.ac.uk, p.l.olivier@ncl.ac.uk

Keywords: human-computer interaction, direct touch interaction, mobile device.

Abstract: We present further developments of our system that allows direction interaction between a camera-equipped hand-held device and a remote display. The essence of this system is the ability to estimate a planar projectivity between the remote display and the displayed image of that display on the handheld device. We describe how to achieve this by matching scale invariant SURF features across the two displays (remote and hand-held). We implement a prototype system and a drawing application and conduct both performance and usability evaluations. The feedback given indicates that our system is responsive, accurate and easy to use.

1 INTRODUCTION

Cameras on hand-held (mobile) devices open up a host of visual interaction techniques and here we focus on interaction with a remote display. Like the mobile device, the remote display is a computer system in itself, such as an intelligent public display, a table-top interaction system, or even just the display of a PC. All that is required is that the camera-equipped mobile device and the remote display can communicate over a wireless channel, such as bluetooth. Since the hardware and software for wireless communication is routinely built into mobile devices, this is generally not an additional cost for our system.

The key concept that we build on is the idea of *registered displays* (Pears et al., 2009). Imagine that we have a mobile device with a camera on one side of the device and a touch display on the other. If we point the camera of the mobile device at a remote display, then we will be able to see some or all of the remote display on the display of the mobile device, depending on the stand-off distance and the field of view of the camera. Assume for now that we can see only a small part of the distant display on the mobile device, such as the file icons within a window. We would like to directly interact with the area of the remote display that we are looking at through the mobile, so that when we touch part of the mobile display (such as the centre of an imaged icon), this touch is projected to exactly the right point (the centre of the same icon) on the remote display. Thus we can view this type of system as 'having a piece of the remote display on a mobile', with which we can interact with. A nice property of this type of system is that, even though only a small part of the remote display may be viewed on the mobile, the user can see the rest of the remote display in their peripheral vision, thus providing context to remote interactions.

Of course, for every point on the mobile display, we must be able to compute the corresponding point on the remote display. This is achieved by estimating the planar projective mapping (homography) between the mobile's image and the remote display and in order to compute this, the mobile's image (or alternatively image features and their positions) must be transmitted to the remote display over a wireless connection. Once we have this mapping, the two displays are said to be 'registered'. Then any point touched with a finger or stylus on the mobile can be transmitted wirelessly to the remote display and projected to its correct corresponding position on the remote display using the estimated homography. An outline of this *registered displays* concept is shown in figure 1.

When the two displays are registered, we can achieve direct interaction with anything on the remote display, by performing actions directly on the mo-

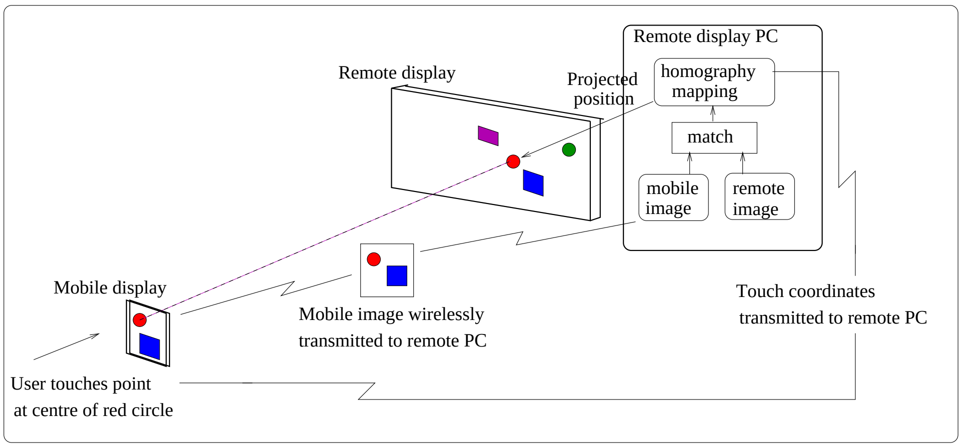
Remote display PC
homography mapping
match
mobile image
remote image
Remote display
Projected position
Mobile display
Mobile image wirelessly transmitted to remote PC
Touch coordinates transmitted to remote PC
User touches point at centre of red circle

Figure 1: Outline of the 'registered displays' concept. The mobile's captured image and 'touched' screen position are transmitted wirelessly to the remote system, which computes the corresponding projected position in the remote display.

bile's image. Direct touch on the mobile's image of the remote display is indistinguishable from directly interacting with the remote display itself. This is hugely beneficial, since the remote display itself does not need to have a touch screen interface, and indeed, one may wish to interact with an remote display that is not physically reachable; for example, consider interacting with a display behind an estate agent's front window, where we could use registered display technology to acquire the details of properties that interest us. Indeed, property details could be sent over the same wireless channel as is used by our registration system.

Equally, we can view this type of system as using the camera as a pointing device. In this viewpoint, when we touch a part of the mobile display with finger or stylus, this point on the mobile display and the optical centre of the camera form a ray in space, which points to the relevant part of the remote display. The estimated planar projective mapping, effectively implements the pointing for us, encapsulating camera position and orientation, camera intrinsic parameters and display intrinsic parameters into a 3x3 matrix.

In this paper, we formulate a markerless display registration system where we propose and evaluate the use of state-of-the-art scale invariant features in order to compute the required plane-to-plane projective mapping (homography).

2 RELATED WORK

The concept of using registered displays via an estimated planar projective mapping is rather new and, to our knowledge, there have only been two systems that directly employ this point-to-point mathematical mapping. The first of these is our previous work (Pears et al., 2008)(Pears et al., 2009) which uses a set of four large green markers on the remote display. The smartphone segments the green markers from the rest of the display and finds each of the four centroids. These centroid positions are passed over the bluetooth link to remote display PC, which then computes the homography between the two displays. Using the estimated homography, any point click on the mobile display can be mapped onto the corresponding point on the remote display. The markers on the remote display are moved and deformed on the remote display such that they are always the same size, shape and position in the smartphone image, thus making them relatively easy to detect. Careful synchronisation is required so that the remote display frame index and smartphone frame index are correctly matched.

The second of these registered display approaches computes the homography by extracting the boundaries of rectangular windows using the Hough transform (Boring et al., 2010). Furthermore, interaction across multiple displays is considered, where display identification is based on the distribution of rectangular windows in the hand held's visible frame.

Early examples of interaction through video include the Chameleon system (Fitzmaurice, 1993) and Object Oriented Video (Tani et al., 1992). Since these early systems, several projects have used handhelds for interaction, for example, collaboration using multiple PDAs connected to a PC in a 'shared whiteboard' application has been demonstrated in the Pebbles project (Myers et al., 1998). Infrared proximity sensors, touch sensors and tilt sensors have been used to initiate and control hand-held interaction (Hinckley et al., 2000). Peephole displays (Yee, 2003) involved pen interaction on spatially aware handheld computers. Many recent proposals for integrating phones and situated displays have sought to use visually-controlled interaction, such as the Mobile Magic Hand (Yoshida et al., 2007), which allows a user to manipulate virtual objects using the measured optical flow in an image sequence, when observing a visual code. Wang and co-researchers developed motion estimation techniques for a camera-equipped phone that derive from full-search block matching, similar to those used by MPEG video encoders (Wang et al., 2006).

3 IMAGE REGISTRATION

The image registration technique that we employ involves scale-invariant feature matching followed by estimation of a plane-to-plane projective mapping (a homography) between the smartphone's camera image plane and the plane of the remote, public display. This projective mapping is described in detail in previous work (Pears et al., 2009). Briefly recapping, if we express a point position on the display in homogenous coordinates as $\mathbf{x}_d = [x_d, y_d, 1]^T$ and a point position on the camera as $\mathbf{x}_c = [x_c, y_c, 1]^T$, then for some scalar λ, their homogenous coordinates are related linearly by a simple 3x3 matrix:

$$\lambda \mathbf{x_c} = \mathbf{H}\mathbf{x}_d \tag{1}$$

This matrix encapsulates 3D camera position, 3D camera orientation, camera intrinsic parameters and display intrinsic parameters, but the important point is that none of these parameters are explicitly necessary to estimate the homography. Rather, we simply estimate (up to scale) the elements within the matrix H, using four known correspondences $(\mathbf{x}_d, \mathbf{x}_c)$ in equation 1. Note that no three points may be collinear. More corresponding points can yield a more accurate estimate of $\mathbf{H}$, using some variant of a least-squares technique.

In earlier work (Pears et al., 2009), we noted that a potential problem with using four coloured markers in order to register displays is that they can be distracting to users and they can cause conflicts when multiple users attempt to interact with the same remote display. Furthermore, there are obvious colour segmentation problems, when the colour of the markers happens to be located in region of similar colour on the remote display. In order to circumvent these problems, we propose scale-invariant feature matching as a means of achieving the required image registration.

The question now is how to find markerless, natural point correspondences between the mobile's image and the remote display's image. As described earlier, the matching problem is made difficult because the hand held's image has been subject to a general planar projective mapping, which may include scale changes, rotations and even general perspective distortion. What we require is a description of a local image patch that does not change (is 'invariant') with respect to typical transformations experienced in the planar projection process.

We decided to evaluate the two most commonly used scale-invariant image descriptors, namely SIFT (Lowe, 2004) and SURF (Bay et al., 2008). Open source MATLAB implementations of these algorithms are readily available. The SIFT implementation used is provided by David Lowe and can be found online (Lowe, 2009). The SURF implementation used is written by Strandmark and can also be obtained online (Strandmark, 2009). In both cases, for performance reasons, the feature detector and descriptor is a compiled C executable which is called by MATLAB. This is a means of overcoming the speed limitations of interpreted MATLAB code.

In a small scale (36 image) evaluation, the most noticeable difference between SIFT and SURF was speed. On average, extracting and describing SURF features was ten times faster than extracting SIFT features. Since the aim is to implement an interactive system with as high a frame rate as possible, SURF was selected as our feature descriptor.

4 PROTOTYPE SYSTEM

Our prototype system employed a mobile webcam operated by the same PC as the display, as shown in figure 2. This is simply a convenient system within which to evaluate our ideas. Of course, it has the disadvantage of not having a touch screen for direct interaction. However, a webcam can be used as a pointing device and pen up/down can be expressed with simple PC keyboard input. Thus we were able to implement a simple remote drawing application, where the central pixel of the webcam points to the pen position on the remote display.

4.1 Background images

Employing images containing extractable features which can be used as a background 'wallpaper' has two main advantages: Firstly, wallpapers cover the whole of the screen. As such, features can be made to be present over the full area of the remote display.

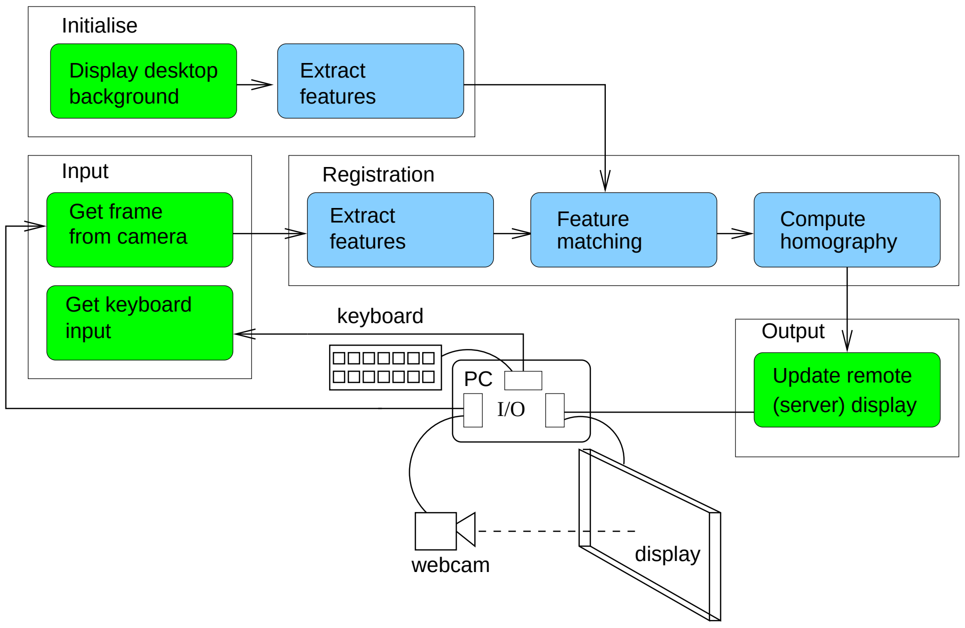

Figure 2: Prototype system outline.

This means that a registration could be computed regardless of camera pose. Secondly, wallpapers are a familiar feature of modern operating systems. If an image is used which was both dense in features but also resembled the type of image a user may select as a wallpaper, the method of presenting extractable features for matching would seem uncontrived. We selected three images, which contained non-abstract, human-understandable images. The images were selected to be varied in their apparent complexity as shown in figure 3

Of course foreground GUI elements, such as windows will obscure parts of the background wallpaper. However, we envisage that the text and icons within windows will often provide sufficient texture to extract scale-invariant features dynamically. Smooth textureless area on the remote display will always be a problem at high zoom levels on the mobile. Part of our ongoing work is to devise a system of natural looking GUI elements that provide sufficient texture at a wide range of zoom scales.

4.2 Feature matching

Our system computes matches between the sensed image on the mobile and precomputed features of the reference image on the remote display by using the 1-nearest neighbour scheme in the feature space. To search for a candidate match for a sensed image feature, the Euclidean distance between its descriptor, D_{si} and all reference feature descriptors D_r are computed. The reference feature with descriptor closest to D_{si} is the most similar and therefore, considered the most likely match. For robustness, however, a further criterion must to be satisfied for the features to be considered a match. An uncertainty' score, u, of the correctness of the match is computed as follows:

$$u(D_{si}, D_{rx}) = \frac{d(D_{si}, D_{rx})}{d(D_{si}, D_{ry})} \qquad (2)$$

Where $d(x,y)$ is the Euclidean distance between vectors x and y and D_{rx} and D_{ry} are the closest and second closest reference descriptor vectors, respectively. If these vectors are equidistant from D_{si}, the uncertainly score has its largest possible value of unity. This indicates that there were two equally close neighbours, so D_{ry} is just as likely to be the corresponding feature as D_{rx}. However, if D_{ry} is much further from D_{si} than D_{rx}, the uncertainly will be very small indicating that no other feature was likely to be a match.

In our system, we specify a threshold uncertainty score and only computed correspondences with an uncertainty below this threshold are returned as matches. In the original SURF implementation (Bay et al., 2008), this uncertainty threshold is set at 0.7 and we use the same value. For the rest of this document, match uncertainty will be discussed in terms of a match confidence score, c, which is defined as:

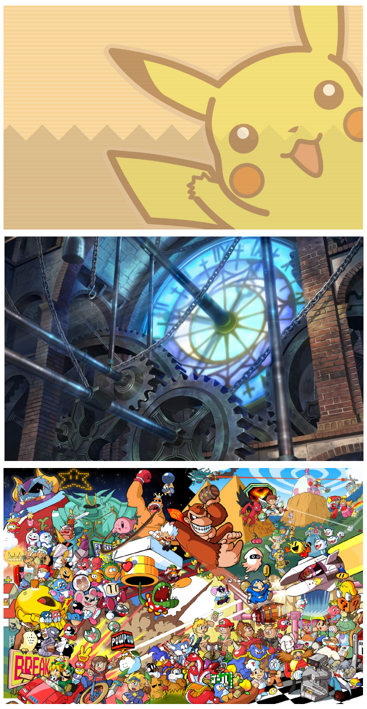

Figure 3: Example wallpapers: (a) low complexity, (b) medium complexity, (c) high complexity with dense features.

$$c(D_{si}, D_{rx}) = \frac{1}{u(D_{si}, D_{rx})} \quad (3)$$

This measure is used to refer to the quality, or strength, of matches, such that the larger the confidence, the stronger the match, since the uncertainty of its correctness is lower. We produce a list of correspondences in descending order of confidence, allowing the strongest matches to be located efficiently. It is still possible that matches returned by this process are incorrect. Detecting incorrect matches on an individual basis, is very difficult. Instead, two different techniques will be used during the computation of the homography to reduce the effect of incorrect matches. The success of this will be measured by the accuracy of the estimated homography in projecting touched points into remote display positions, since this can be automated more easily than measuring match correctness directly.

4.3 Computation of the homography

Only four feature matches (correspondences) are required to determine H, subject to the constraint that no three are collinear. However, for our complex wallpaper image, typically several hundred (around 200-600) feature matches are made per frame. Our aim was to generate a homography estimate that is both fast and accurate. To this end, we implemented and tested two systems.

In **system 1**, we used the strongest 50% of correspondences, those with the largest match confidence, to estimate the homography. This improves the speed of the least-squares homography estimate and, assuming weaker matches are more likely to be incorrect, this should improve the proportion of correct matches and hence improve the homography estimate.

In **system 2**, we used a sample consensus approach, motivated by the RANSAC approach (Fischler and Bolles, 1981). The principle is (i) to calculate a strong initial homography estimate using a minimal set of four correspondences, (ii) to identify the remaining correspondences which agree with this estimate, and finally (iii) use this larger set to compute a new, improved least-squares homography estimate.

Our algorithm takes two lists, P_r and P_s, such that $P_r(i)$ is a point in the reference image, computed as corresponding to the point in the sensed image given by $P_s(i)$. These lists are ordered in descending order of confidence, as returned by the feature matching component. First, the four strongest, non-collinear correspondences are identified and used to compute an estimate homography, H_0. By using a minimal set of the strongest available matches, the chance of this estimate being made using an incorrect correspondence is minimised. However, local inaccuracy is possible due to noise in the image leading to imperfect measurement of the feature positions in the images. A new list, P'_s is computed such that

$$P'_s(i) = H_0 P_r(i) \quad (4)$$

This list contains the points in the sensed image corresponding to each point in P_r, as suggested by the initial estimated homography. The Euclidean distance between each pair $(P_s(i), P'_s(i))$ is calculated. If this distance is smaller than some threshold, the correspondence containing $P_s(i)$ is classed as an inlier, otherwise it is an outlier. All of the inlier correspondences are then used to compute the final homography estimate, H, using least squares. Using a large set of points minimises noise-related error and only using points which agree with the initial estimate largely excludes incorrect correspondences.

5 PERFORMANCE EVALUATION

A test cycle was designed to test the performance of the underlying registration system; not the usability of the system as a whole, which is discussed in section

6. As such, the initial testing framework was designed to measure speed and accuracy of display registration using the high complexity wallpaper (c) in fig. 3.

In order to ensure testing was fair and repeatable, live input from the client device was not used. Replicating camera movements manually is unreliable and would affect the results gathered in testing. Instead, the hand-held webcam was used to record four video sequences. These videos were used as input, allowing tests to be run on identical data. Each of the four videos was designed to represent one of the possible geometric transformations the system would need to deal with. These are:

1. **Pan.wmv:** The camera is held with the image plane parallel to the display and translated (in-plane) from left to right at a constant distance of 30cm from the display.
2. **Zoom.wmv:** The camera is held central and parallel to the display and moved from a distance of 10cm to 50cm from it. This tests out-of-plane translations (or equivalently, zooming out).
3. **Rotate.wmv:** The camera is held central and parallel to the display at a distance of 30cm, while it is rotated (in-plane) through 180 degrees.
4. **Skew.wmv:** The camera starts by pointing towards the centre of the display at a distance of 30cm. The camera then follows a roughly circular arc through 30 degrees, rotating about the display's y-axis, until it is 5cm from the screen and at a 60 degree angle to the screen.

Given an input video, our evaluation framework computes the homography between the frames of the video and the background image, exactly as the live system would. We record timing data and the computed homography for each frame processed and save this data to disk for analysis. We record the time it takes for each component of the system to execute as well as the time taken for each frame to be processed.

To facilitate the measuring of accuracy, we manually selected the pixel in the reference image corresponding to the pixel at the centre of each frame in each of the test videos, as accurately as possible. These manually selected points correspond to where the cursor should be on the remote display if the system was being used as a pointing system. These points serve as the ground truth position of the cursor and allow the accuracy of the point projected on to the remote display to be measured. More specifically, for any frame of a test video, upon computing the homography, the cursor position (central camera pixel) projected by this homography can be determined. This can be compared to the manually determined (ground-truth) cursor position for the frame to give an indication of accuracy. Accuracy is measured as the Euclidean distance, in pixels, between the computed and manually selected cursor positions. This measure is computed for every frame of every video and saved to disk for analysis.

We evaluated the accuracy of system 1, where the best 50% of feature matches are used to compute the homography, and system 2, which employs sample consensus. An overview of average performance across all videos for the two systems is shown in figure 4. The sample consensus approach (system 2) shows superior performance, particularly in the difficult 'skew' video.

Examining the detail of the accuracy on a frame-by-frame basis, we see that for (i) translating, (ii) zooming and (iii) rotating in-plane, the error is acceptably small and always within five pixels. However, in the 'skew' video sequence as the angle of camera approaches 30 degrees from the remote display normal, the errors can become large, particularly in system 1. This is because SURF features are not invariant to general projective transformations and so the quality of feature matching deteriorates.

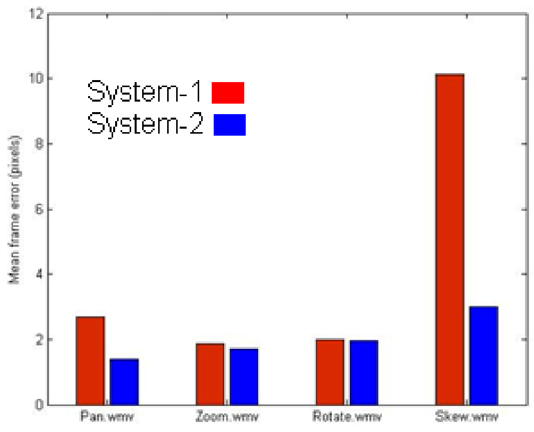

Figure 4: Mean pixel error results for the four movies.

On examining the time to process a frame, there were no significant differences between any of the videos. We also noted that the mean frame rate, across all frames, is 0.64 frames per second. This is clearly too slow for a direct interaction device; the cursor will only update every 1.56 seconds which would result in a very low level of usability. Within our timing results, we found that feature extraction took on average 17.8% of the time, feature matching took 74.4% of the time, whilst computing the homography took only 7.8% of the time.

6 USABILITY EVALUATION

In our system, both responsiveness and accuracy are preferred. However, this is not currently possible and some trade-off has to be made. In order to determine where a good balance between speed and accuracy lies, a user test was conducted. This was also used to quantitatively measure perceived usability.

We evaluated three techniques to improve the system update rate: (i) bounding the time over which feature matching can take place; (ii) reducing the sensed image resolution on the camera; (iii) using a wallpaper with a lower density of features.

Four different system configurations were used, each with a different speed-accuracy trade off. Each user conducted the same task with the system in each configuration and asked to provide feedback on their experiences. The order in which the user tested each configuration was random, to reduce bias being introduced as the user learned the task. The details of the four configurations are as follows:

- **System A: Maximum Accuracy.** High complexity background in fig. 3(c), unbounded match time, 640x480 sensed image. Mmean error per frame = 1.96, frames per second = 0.59
- **System B: Accuracy Biased.** High complexity background, 350ms bounded match time, 640x480 sensed image. Mean error per frame = 3.47, frames per second = 0.87
- **System C: Speed Biased.** Medium complexity background in fig. 3(b), unbounded match time, 640x480 sensed image. Mean error per frame = 2.54, frames per second = 1.89.
- **System D: Maximum Speed.** Medium complexity background, unbounded match time, 320x240 sensed image. Mean error per frame = 16.12, frames per second = 3.07.

The stated mean error per frame is calculated over all frames for the relevant test videos while the frames/second measure is taken from the application itself and so accounts for all overheads related to updating the display to show the user input.

The task involved drawing and a single instance of the task was made up of three units. For each unit, a simple shape was shown on the remote screen, over the top of the background. Each of the three shapes was designed to test a specific drawing action. A triangular shape encouraged the user to track straight lines and sharp corners, a circle encouraged the user to track a continuous curve and a set of dashed lines encouraged the user to start and stop line tracking.

The user was asked to trace the shape by drawing over the top of it, as *"quickly and accurately as possible"*. Before the start of each task, the participant was given thirty seconds to practice drawing with no shape to trace. This gave them time to familiarise with the current configuration before the task begun. Upon completing the task, they were asked to fill in a form, rating their experience of conducting the task against three criteria: (i) responsiveness (how quickly movements of the camera were reflected on screen); (ii) accuracy (how accurately movements of the camera were reflected on screen); (iii) usability (how simple it was to complete the task). All scores were to be between 1 (low) and 10 (high). Participants were also asked the deliberately open-ended question: *"What would have made completing the task easier?*. This question allowed the user to talk more freely about their experience with the system, allowing us to determine which factors were most important to them.

The test described above was undertaken by six volunteers. The participant was instructed to conduct each unit of the test as quickly and as accurately as possible, but was told that they could rest between each unit and between each task for as long as they wished. They were supervised in their first practice period to ensure that they had understood the instructions. The participant was then left alone to complete the test in their own time.

Responsiveness results 1.83, 2.33, 5.50 and 7.67 were the mean scores (max 10) given by participants when asked to rate *'how quickly movements of the camera were reflected on screen* for each configuration A to D respectively. These results were as expected. The reported responsiveness corresponds to the frame rate of the configurations.

Accuracy results 2.83, 4.33, 6.50 and 7.50 were the mean score given by the participants when asked to rate *'how accurately movements of the camera were reflected on screen* for each configuration A to D respectively. It should be noted that in the automated testing, configuration A was the most accurate, while configuration D was the least accurate. However, in the user tests, configuration D received a higher average rating than A.

Ease of use results 4.0, 4.50, 4.50 and 8.33 were the mean scores given by the participants when asked to rate *'how simple it was to complete the task* for each configuration A to D respectively. Configuration D received a much higher mean rating for this criterion.

Qualitative results and comments Figure 5 shows sample drawings by participants, using configuration D (top images) and configuration A (bottom images). While not perfect, the drawings with configuration D show no spikes due to registration error and the beginning, ends and general position of most lines appears to be accurate.

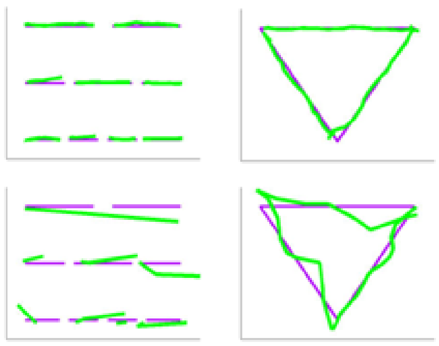

Figure 5: Sample drawing results: system D (top) and system A (bottom).

All six participants said that if the response time of configuration A was faster, the task would have been easier to complete. Similar remarks were made by three or more of the participants for configurations B and C. Only two of the six participants said that a faster response time would have made the task easier when using configuration D. Accuracy was only explicitly mentioned by two participants. One of these participants made the statement about configuration C and the other said improving the accuracy of the crosshair would have made the test easier on all configurations. The background images were mentioned by two participants. Specifically, it was noted that the colourfulness of the most complex background made it difficult to locate the crosshair or the lines to be traced, making the task harder.

7 CONCLUSIONS

Development of a working prototype system has shown that the extraction and matching of scale-invariant features provides an effective route to markerless mobile/display interaction. Usability testing was undertaken and our prototype system in configuration D received high scores for responsiveness, accuracy and ease of use. During development it was identified that a responsive system (high frame rate) gives a greater level of perceived usability than a less responsive, but more accurate one. Overall the work presented here has demonstrated the possibility of a marker-less display registration system, which can be built upon to create rich, direct interaction applications via camera-equipped mobile devices, such as smartphones.

REFERENCES

Bay, H., Ess, A., Tuytelaars, T., and Gool, L. V. (2008). Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 100(3):346–359.

Boring, B., D.Baur, Butz, A., Gustafson, S., and Baudisch, P. (2010). Touch projector: Mobile interaction through video. In *ACM Conference on Human Factors in Computing Systems (CHI 2010)*, pages 2287–2296.

Fischler and Bolles (1981). Random sample consensus: a pardigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:381–395.

Fitzmaurice, G. W. (1993). Situated information spaces and spatially aware palmtop computers. *Communications of the ACM*, 36(7):39–49.

Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. (2000). Sensing techniques for mobile interaction. In *13th Annual ACM Symposium on User interface Software and Technology*, pages 91–100.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110.

Lowe, D. G. (2009). Demo software: Sift keypoint detector. http://www.cs.ubc.ca/ lowe/keypoints/.

Myers, B., Stiel, H., and Gargiulo, R. (1998). Collaboration using multiple pdas connected to a pc. In *Proc. 1998 ACM Conference on Computer Supported Cooperative Work*, pages 285–294.

Pears, N. E., Jackson, D. G., and Olivier, P. L. (2009). Smartphone interaction with registered displays. *IEEE Pervasive Computing*, 8(2):14–21.

Pears, N. E., Olivier, P. L., and Jackson, D. G. (2008). Display registration for device interaction. In *Proc. 3rd Int. Conf. on Computer Vision Theory and Applications (VISAPP'08)*.

Strandmark, P. (2009). Demo software: Surfmex. http://www.maths.lth.se/matematiklth/petter/surfmex.php.

Tani, M., Yamaashi, K., K.Tanikoshi, Futakawa, M., and Tanifuji, S. (1992). Object-oriented video: Interaction with real-world objects through live video. In *ACM Conference on Human Factors in Computing Systems (CHI 1992)*, pages 593–598.

Wang, J., Zhai, S., and Canny, J. (2006). Camera phone based motion sensing: Interaction techniques, applications and performance study. In *19th Annual ACM Symposium on User interface Software and Technology (UIST'06)*, pages 101–110.

Yee, K. (2003). Peephole displays: Pen interaction on spatially aware handheld computers. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 1–8.

Yoshida, Y., Miyaoku, K., and Satou, T. (2007). Mobile magic hand: Camera phone based interaction using visual code and optical flow. In *Human-Computer Interaction Part II, LNCS 4551*, pages 513–521.