Augmented Reality Camera Tracking with Homographies

Simon J.D. Prince, Ke Xu, and Adrian David Cheok
National University of Singapore

To realistically integrate 3D graphics into an unprepared environment, camera position must be estimated by tracking natural image features. We present a fast, robust tracking system based on computing homographies.

Despite the increasing sophistication of augmented reality tracking technology, a recent survey of AR notes that "tracking in unprepared environments remains an enormous challenge."[1] Although camera-based tracking for AR environments using fiducial (reference) markers has been highly successful,[2] there are many scenarios in which we might want to add virtual content without preparing the environment by inserting artificial markers. A prototypical example is the labeling of geographical features for navigation in an outdoors AR setting. Here, we must track the camera using natural scene features alone. In the past, this problem has been approached in one of two distinctly different approaches.

The geometric approach reconstructs the full 3D camera motion based on mathematical constraints between feature positions in multiple images. For example, the fundamental matrix is an algebraic entity that embodies the information about camera movement between two frames of the same static scene. (See the literature on the fundamental matrix calculation—and the subsequent extraction of motion parameters—especially Hartley and Zisserman's theory review.[3]) Commercial movie studios typically use these and similar methods to introduce computer graphics into film sequences. However, such algorithms are unsuitable for real-time applications. These methods, which must generally batch-process the entire data sequence at once (an operation known as global bundle adjustment), are too time-consuming for interactive applications.

The second approach to natural-feature tracking is image based. Instead of estimating the full 3D camera motion, the camera tracks and labels features across a sequence of images by measuring the 2D image motion. Such "optical flow" methods are faster than the geometric approach but, because they can't estimate 3D camera motion, they can't be used to introduce 3D augmentations into scenes. Moreover, optical-flow tracking is unstable and requires heavy regularization to produce a reasonable solution.

We offer an intermediate approach we developed to the tracking problem that encompasses the advantages of both geometric and image-based tracking. We apply this technique to cases where feature positions in adjacent frames of an image sequence are related by a homography, or projective transformation. We describe this transformation's computation and demonstrate several applications. First, we use an augmented notice board to explain how a homography, between two images of a planar scene, completely determines the relative camera positions. Second, we show that the homography can also recover pure camera rotations, and we use this to develop an outdoor AR tracking system. Third, we use the system to measure head rotation and form a simple low-cost virtual reality (VR) tracking solution.

Homography overview

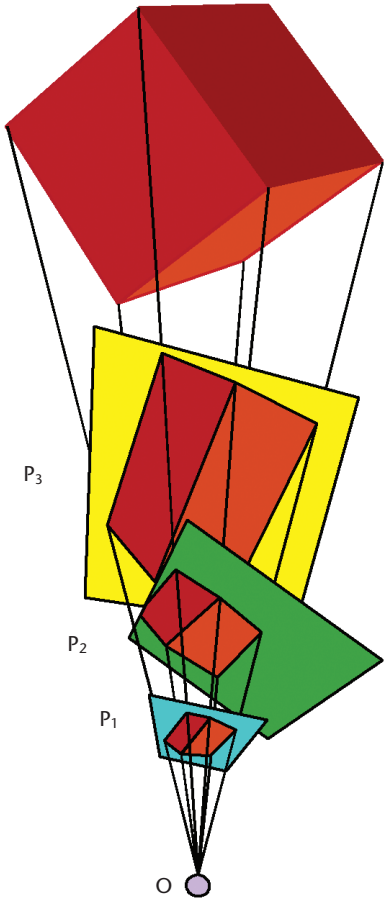
A *homography* is a one-to-one mapping between two images, which is defined by only eight parameters. This model exactly describes the image motion between two frames of a video sequence when

- the camera motion is pure rotation, or
- the camera is viewing a planar scene.

Usually, feature displacement between two images depends on both the camera movement and the camera's distance from the feature. A simple parameterized mapping is therefore not possible. However, in many circumstances, the homography represents a good approximation of the true image flow, particularly when the image structure is near planar, or the camera movement is small and the scene structure is mostly distant.

A number of previous systems have applied this information. For example, Uenohara and Kanade's[4] system achieves medical-image registration based on five coplanar points. However, their system's disadvantage

0272-1716/02/$17.00 © 2002 IEEE



1 Geometric representation of a planar projective transformation or homography. The images on different camera planes that cut the same ray bundle are related by homographies.

is that it isn't robust if any points are obscured or not detected. Another example is Simon and Fitzgibbon's[5] offline system for introducing virtual content into stored video sequences.

Automatic computation of a homography

Consider a set of points in the first image of a sequence with homogeneous coordinates (x_i,y_i,z_i), which are known to map to a set of points in the second image, (x'_i,y'_i,z'_i). The relationship between the two images is a homography if the following equation holds:

$$\begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$$

In other words, the homography, **H,** maps coordinate **x** to coordinate **x′**. Note that these are homogenous coordinates—each point on the screen is treated as a ray through the camera center. We find the actual image position by dividing the first and second components by the third. The homography is, therefore, a simple linear transformation of the rays passing through the camera center. Roughly speaking, the homography can encompass rotations, scaling, and shearing of the ray bundle.

Figure 1 presents another way to consider this concept. Instead of transforming the rays, we can equivalently transform the camera plane onto which the rays are projected. The diagram shows the cube's rays passing to the camera center, O. The three image planes, P_1, P_2, and P_3, intersect these rays in three different ways. The resulting images are related to one another by projective transformations or homographies. A characteristic property of homography is that it always maps a straight line to another straight line, but parallelism is not necessarily preserved.

A homography relating two images

Although the matrix describing the homography contains nine elements, it's ambiguous up to scale—the use of homogeneous coordinates means that any multiple of the homography will have the same effect. Consequently, because there are only eight independent elements, we can measure the homography relating two images using any four general point correspondences. Each point correspondence generates two linear equations.

To solve for the elements of **H**, we note that **x′** and **Hx** are, by definition, rays pointing in the same direction. Their cross product is equal to zero:

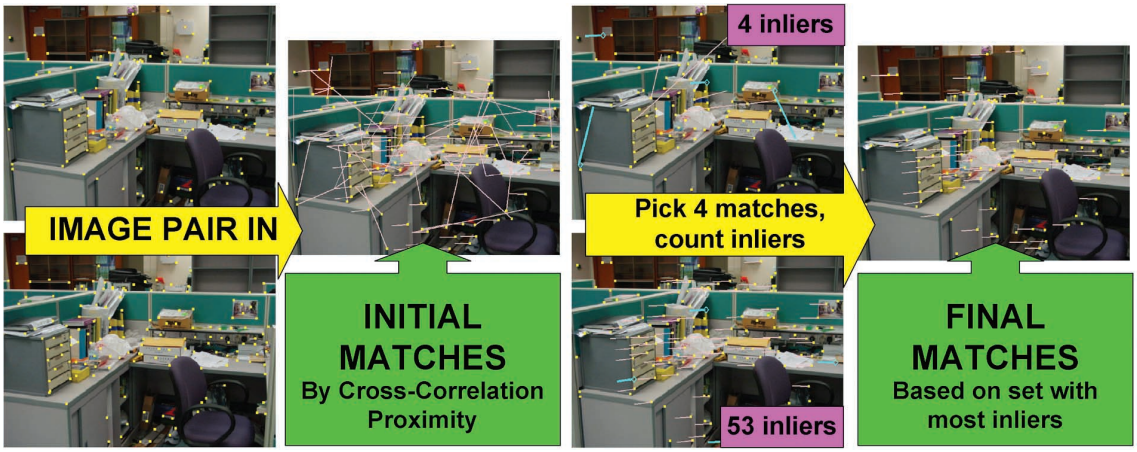
$$\mathbf{x}' \times \mathbf{H}\mathbf{x} = \begin{bmatrix} y'\mathbf{h}_3\mathbf{x} - z'\mathbf{h}_2\mathbf{x} \\ z'\mathbf{h}_1\mathbf{x} - x'\mathbf{h}_3\mathbf{x} \\ x'\mathbf{h}_2\mathbf{x} - y'\mathbf{h}_1\mathbf{x} \end{bmatrix} = 0$$

where $\mathbf{h_3}$ is the third row of the homography matrix, **H**. This set of equations provides two independent linear constraints on the components of **H**. If we find four general point correspondences, we can provide eight equations to solve for the eight unknowns of the homography. Generally, we aim to find more point correspondences and calculate an overdetermined least-squares solution. The error in the point positions is linear, but these terms are quadratic in the set of equations. Ideally, we should use this solution as an initial estimate in a subsequent nonlinear minimization of the Euclidean projection error.

Choosing correct matches

How might we exactly identify four corresponding points? We use a Harris corner detector to identify 50 to 60 points of interest in each image.[6] This identification establishes image positions where the intensity gradient varies significantly in both the *x* and *y* directions. We constrain these points to be at least 20 pixels apart to ensure that we get a good spread of corners across each image. Given 50 corners in each image, we have 50 potential matches for each of the 50 corners in the first image. However, some of these 2,500 pairs are more probable than others. We predict each corner's movement from the first image to the second. If the potential match in the second image isn't close to this predicted position, we disregard it. The prediction can be based on previous estimates of the homography (such as a Kalman filter formulation), or it might come from other tracking devices such as accelerometers.

We select the initial matches from the remaining possibilities by examining the similarity of the local 15×15 pixel regions surrounding the corners. For each point in the first image, we find the corner in the second image



2 Robust calculation of a homography between two images. Corner points are identified in the two images (yellow dots). We choose initial matches based on the similarity of the areas around these corners and on prior knowledge about the likely match direction (pink lines indicate corner vector to matched corner in other image). This initial set contains many incorrect matches. We pick four matches (blue lines) and calculate the associated homography. We then count the number of other matches that are in agreement (inliers are pink lines) and repeat this procedure. We choose the estimate with the most support and recalculate the homography using all of the inliers.

with the highest local cross-correlation. We now have an initial set of matches (see Figure 2). However, it's clear that some of these are erroneous and will render a brute-force least-squares solution ineffective. How, then, to sort the wheat from the chaff? We employ a robust statistical method known as random sample consensus (Ransac)[7] to establish the correct matches.

The Ransac procedure involves choosing a subset of four matches at random and calculating a homography. We note how many of the other matches are in close agreement to the predictions of the homography (inliers). We repeat this process a number of times, keeping track of the mapping that had the greatest support. Finally, we estimate the homography based on the inliers. In pseudocode,

```
MaxInliers = 0;
For (n = 1 to nIterations)
   Choose random subset of four matches
   Calculate Homography
        Calculate how many of the other
          matches agree (inliers)
   If (inliers>maxInliers)
      bestHomography = thisHomography
Recalculate Best Homography from All
     inliers
```

The right-hand side of Figure 2 shows the results of this procedure. The final point matches are in close agreement, indicating that image movement is predominantly horizontal. These methods are well established in the computer vision literature.[3]

Implementation details

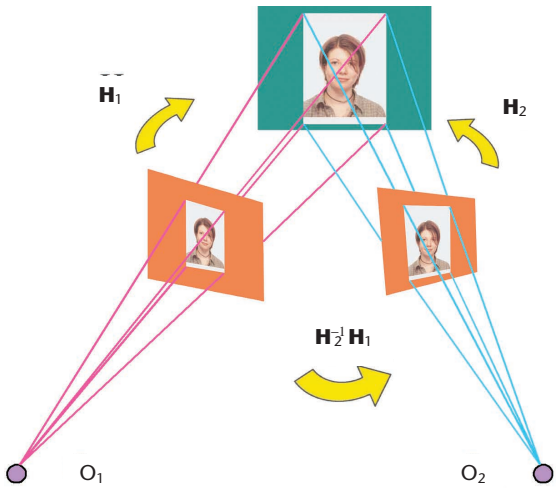
This system lets us calculate homographies at 25 Hz—the capture rate of the PAL video stream—between 320 × 240 pixel grayscale images on a standard desktop PC. Typically, the system identifies 50 to 60 corners per image, of which more than 80 percent are inliers to the final solution, depending on the overlap of the two images. If less than 40 percent of corners are inliers, we consider the calculation to have failed.

We apply a number of heuristics to increase the system speed and accuracy—we repeat the Ransac sampling up to 70 times, but we immediately accept the solution if it has greater than 75 percent support and if we've already completed more than 15 iterations. Using this criterion, the majority of frames complete prematurely. To increase the solutions' average quality, we ensure that the initial four points in the first image are spatially separated by at least 60 pixels. We also reject homographies that are near singular because these map the entire first image to a point or line in the second image. Singular matrices map the entire first image to a line or point in the second image.

We performed some simple tests to assess the reliability and accuracy of this calculation. For a static camera in an indoor environment, we successfully calculated the homography for 500 of 500 test frames in a 20-second sequence. In each case, we measure the homography to the previous frame. We use this to predict where the center point of the previous image maps to in the current image. Because the camera is static, we assume that it ought to remain in the same place. We found that the mean deviation was less than 1 pixel per frame.

We also investigated the system's robustness under pure rotations around a vertical axis. Homographies in our tests were always computed with a greater than 90 percent success rate even up to 25 degrees of rotation between the two images. Accuracy was comparable to the fixed case for moderate rates of rotation, but it can decrease as the image overlap—and the contributing data—decline with increased rotation.

These results are for matching two static but rotated



(a)

(b)

3 (a) Any two views of a planar scene are related by a homography. (b) We exploit this to superimpose 2D and 3D graphical content onto a real notice board.

images and don't consider the effects of motion blur that might occur during real rotations. Although our system is already extremely accurate, it's possible that it might improve if the estimation were embedded in a Kalman filter with a prior model for the camera motion, or combined with data from other sensors.

Applications

We applied our approach for camera tracking in three different scenarios.

Planar scene structure

The homography completely describes the relationship between any two ideal images of a planar structure (such as a notice board). Figure 3a shows why this is the case, in which two cameras view the same board from different angles. Consider the rays projecting outward from camera center, O_1. Because the notice board and the image plane 1 both intersect the same ray bundle, they must be related by a homography, which we denote $\mathbf{H}_1$. A similar argument applies to the relationship between the notice board and image plane 2, which the homography $\mathbf{H}_2$ describes. Because homographies form a closed group under multiplication, the two images of the board must be related by a third homography $\mathbf{H}_2^{-1}\mathbf{H}_1$.

We can introduce 2D virtual content onto a real planar scene, as follows, in introducing a virtual photo onto a notice board. We store a picture of the surface and note the positions of the corners of our desired virtual photo (see Figure 3b). For each frame, we calculate the homography from the current image of the stored image. We use this calculation to predict the current positions of the photo's corners, which can then be drawn onto the notice board in real time. In fact, if we know the camera calibration matrix and have prior knowledge of the relative position of some points on the board, we can calculate the camera's full 3D transformation relative to the planar surface. Indeed, existing tracking schemes based on fiducial markers calculate this 3D transformation by measuring the corner positions of a square pattern of known size (see Kato et al.[2]). This is clearly equivalent to tracking the corners of our 2D photo and allows the introduction of 3D objects into the image, such as the cube in Figure 3b.

We can establish a direct solution for the 3D transformation from the homography itself. (See Sturm[8] and Zhang[9] for details. Also, Simon implemented a system, which didn't run in real time, based on this approach.[5]) A simple way to initially position the board's 3D virtual content is to place a fiducial marker only in the stored image. The matching is sufficiently robust to overcome this discrepancy in the surface's contents. Using these techniques, we produce tracking performance based on natural features that's qualitatively equivalent to that produced by Kato and Billinghurst's fiducial marker tracking system.[2]

One potential problem is that because the cross-correlation stage isn't orientation invariant, it will fail if we try to match to the stored frame after significant rotation around the z-axis. Although this situation is rare because it's difficult for users to move their head this way, we've developed a simple solution. We warp the cross-correlation region in the current image by a prior estimate of the homography mapping to the stored image. Currently, we take the homography calculated for the previous frame as a prior estimate. If the previous frame was rotated about the z-axis relative to the stored frame, then the cross-correlation window around each point in the current frame will be rotated accordingly. In this way, we can indefinitely achieve the approximate alignment of regions surrounding the corners with those in the stored frame.

Pure camera rotation

Figure 4a shows why a homography can describe, in addition to planar structure, pure camera rotation. The left image shows a camera viewing a 3D cube. When the camera rotates, the image plane simply cuts the rays at a different angle. Rotations thus form a subset of the transformations described in Figure 1. This knowledge lets us apply natural feature tracking, based on homographies, to the problem of registering information in outdoor scenes. Such a problem has previously been attacked through various methods, including inertial trackers, 2D computer vision, and hybrid approaches.[10] We argue that outdoor tracking requirements are frequently satisfied by calculating a homography to a stored reference frame. Because most objects in this setting are distant, we can con-

sider camera movement to be approximately a fixed rotation.

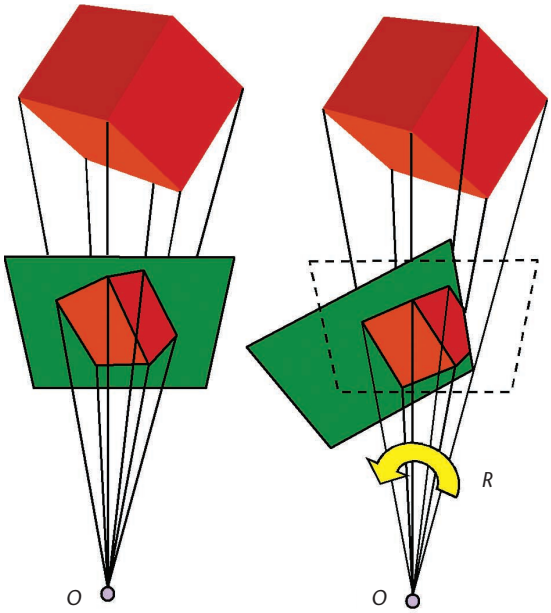
Our outdoor AR tracking system operates as follows: We store a reference image or mosaic of a scene and note the correct position for feature labels, as in Figure 3b. We calculate a homography between the current frame and this reference image, then use it to map the label position to the current frame. In practice, we needn't store reference images, only the corner positions and surrounding regions. We can form a large dictionary of reference data, which is indexed by cruder measurements from GPS (Global Positioning System) and compass readings.

VR tracking system

Our third application is a VR tracking system that is also based on measuring pure camera rotation. Billinghurst et al.[11] drew attention to the use of AR as a "transitionary" interface between the real environment and VR. They presented a "magic" book as in Figure 5 that lets users interact at any of three levels. The book can be read normally or viewed through a head-mounted display to reveal 3D augmentations to the pages. Alternatively, the user can enter VR mode and step inside the scenes depicted in the book. Unfortunately, this transition requires two separate tracking infrastructures—the AR mode uses a head-mounted camera to track fiducial markers, whereas the VR mode requires a more general VR tracking system. We can eliminate the need for this second system by tracking natural features in the head-mounted camera. For each pair of frames, we estimate the homography and then find the "closest" Euclidean rotation to this homography (see Zhang and Faugeras for another possible method based on a direct linear solution for the rotation[12]). We chain these rotations together to form a current estimate of the user's view of the VR scene. On entering VR mode, users can examine the scene simply by rotating their head, with no need for a second tracking system.

System assessment

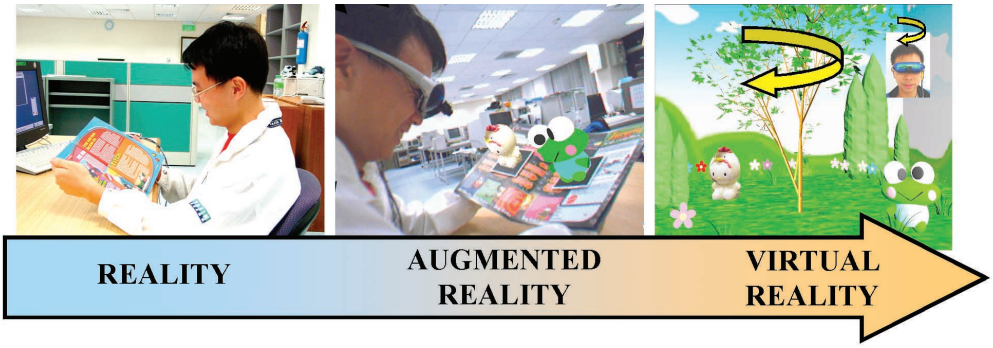
In addition to simplicity and ease of calculation, the system we've described has three major advantages over most competing algorithms. First, the homography calculation is robust. Notice that, in Figure 3, the system places the photo correctly even when one of the



(a)

(b)

4 (a) Pure rotation (*R*) about the camera center (*O*) is a special case of the general projective transformation. (b) Superposition of navigational information in a mobile AR setting.



5 Augmented reality (center) can be used as a transitional interface between the real world (left) and virtual environments (right). Natural feature tracking can smooth this transition—head rotation in VR can be estimated without the need for further hardware.

One of the advantages of our system is that it provides a simple method for storing information about natural scenes.

authors obscures part of the notice board—he is neither planar nor present in the reference frame but calculation succeeds nonetheless. Corners in this image that are part of his body will not find support across the rest of the images, and subsequent matches will be rejected accordingly.

Similarly, the system is robust under small translations for outdoor registration as long as the majority of the scene contents are distant. In fact, as anyone who has tried to estimate the general geometrical relationship between two images (that is, the fundamental matrix) can attest, a homography represents a reasonable model in many real-world situations. The movement of image points resulting from small lateral translations and moderate translations along the camera axis is well described by the homography. In these circumstances, it's possible to improve the virtual object's positioning by more heavily weighting matches near the virtual object. The system is thus well suited for 2D textual augmentations (see Figure 4) even when the camera movement isn't a pure rotation.

Second, the system generates absolute estimates of the virtual content position—we store a fixed set of information about the world and match this on each frame. Compare this to tracking optical-flow features over a series of images or to measuring the change in position using inertial trackers. However accurate the measurement is, errors are bound to accumulate over a number of frames, and the virtual object will drift from the correct position. Any system based solely on iterative relative position estimation can't, even in principle, solve the tracking problem.

Third, the homography calculation is extremely fast. We can track natural visual features at camera-capture frame rates on a standard desktop PC—to the best of our knowledge, the first computer-vision-based AR tracker to do so. We could further improve calculation speed by using measurements from inertial sensors to provide the homography's initial estimate—this reduces the number of possible matches and therefore the overhead of the cross-correlation calculation. If more accurate measurements were available, we could avoid the cross-correlation stage altogether and simply form our initial matches based on their agreement with information from other sources.

A further advantage of the system is that it provides a simple method for storing information about natural scenes—we store a list of corner directions and the local image regions around the corners, which is a highly efficient representation. Because we indexed these lists using GPS and compass readings, the current view is always compared to the correct stored data.

The most obvious weakness of our system is that it relies on the ability to find and track stable corners in the incoming video stream. The system won't perform well if the camera view contains large blank walls or highly dynamic scenes such as crowds. Such problems are ubiquitous in current vision-based tracking algorithms. The likely solution is to combine the system with data from other tracking modalities.

Conclusion

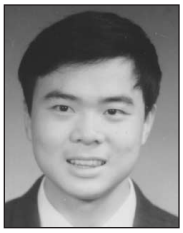
As we've shown, the general problem of tracking camera motion based on visual information alone is a difficult one. Camera pose recovery is intimately coupled with the problem of estimating the 3D scene structure from multiple camera views. Future developments in these areas will rely on improvements in how computer models of the world are represented and how to correlate these models with incoming data from a video stream. ■

References

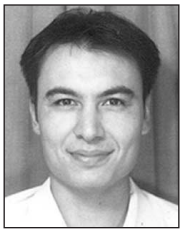
1. R. Azuma et al., "Recent Advances in Augmented Reality," *IEEE Computer Graphics and Applications*, vol. 21, no. 6, Nov./Dec. 2001, pp. 34-37.
2. H. Kato and M. Billinghurst, "Marker Tracking and HMD Calibration for a Video Based Augmented Reality Conferencing System," *Proc. Int'l Workshop on Augmented Reality* (IWAR 1999), IEEE Computer Soc. Press, Los Alamitos, Calif., 1999, pp. 85-94.
3. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge Univ. Press, Cambridge, UK, 2000.
4. M. Uenohara and T. Kanade, "Vision-Based Object Registration for Real-Time Image Overlay," *Int'l J. Computers in Biology and Medicine*, vol. 25, no. 2, Mar. 1995, pp. 249-260.
5. G. Simon, A.W. Fitzgibbon, and A. Zisserman, "Markerless Tracking using Planar Structures in the Scene," *Proc. Int'l Symp. Augmented Reality* (ISAR 2000), IEEE Computer Soc. Press, Los Alamitos, Calif., 2000, pp, 120-128.
6. C.J. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Proc. 4th Alvey Vision Conf.*, Univ. Printing Unit, Sheffield, UK, 1988, pp. 147-151.
7. M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Readings in Computer Vision: Issues, Problems, Principles and Paradigms,* M.A. Fischler and O. Firschein, eds., Morgan Kaufmann, Los Altos, Calif., 1987, pp. 726-740.
8. P. Sturm, "Algorithms for Plane Based Pose Estimation," *Proc. Computer Vision and Pattern Recognition Conf. (CVPR 2000)*, IEEE Computer Soc. Press, Los Alamitos, Calif., 2000, pp. 1010-1017.
9. Z. Zhang, "Flexible Camera Calibration by Viewing a Plane from Unknown Orientations," *Proc. Int'l Conf. Computer Vision* (ICCV 1999), IEEE Computer Soc. Press, Los Alamitos, Calif., 1999, pp. 666-673.
10. S. You, U. Neumann, and R. Azuma, "Orientation Tracking for Outdoor Augmented Reality Registration," *IEEE Computer Graphics and Applications*, Nov./Dec. 1999, pp. 36-42.
11. M. Billinghurst, H. Kato, and I. Poupyrev, "The Magic Book: An Interface that Moves Seamlessly between Reality and

Virtuality," *IEEE Computer Graphics and Applications*, vol. 21, no. 3, May/June 2001, pp. 6-8.
12. Z. Zhang and O. Faugeras, *3D Dynamic Scene Analysis*, Springer Verlag, Berlin, 1992, pp. 61-61.

Simon J.D. Prince *is a Research Fellow in the Dept. of Electrical and Computer Engineering, National University of Singapore. His research interests include visual neuroscience, nonstationary inverse problems, mobile augmented reality, and image-based rendering. Prince has a BS in psychology and an MRes in computer science, both from University College, London, and a PhD in psychology from the University of Oxford, UK.*



Ke Xu *is an M. Eng. student in the Dept. of Electrical and Computer Engineering, National University of Singapore. His research interests include augmented reality, human-computer interaction, and computer vision. Xu has a BS in electrical and computer engineering from the National University of Singapore.*



Adrian David Cheok *is an assistant professor in the Dept. of Electrical and Computer Engineering, National University of Singapore. His research interests include mixed-reality and wearable computers, industrial electronics, embedded systems, fuzzy logic, and soft computing. Cheok has a BS and a PhD in electrical engineering from the University of Adelaide. He is currently chairman of the IEEE Systems, Man and Cybernetics Chapter, Singapore.*

Readers may contact Simon Prince at the Dept. of Electrical and Computer Engineering, National University of Singapore, 4 Engineering Dr. 3, Singapore 113576, email elesp@nus.edu.sg.

For further information on this or any other computing topic, please visit our Digital Library at http://computer.org/publications/dlib.

IEEE ComputerGraphics AND APPLICATIONS

2003 Editorial Calendar

January/February

Web Graphics

With the popularity of the Internet, we're seeing a migration of traditional applications to run on the Web environment and a growing demand for more powerful Web-based applications. Fused by the increasing availability and dramatic reduction in the cost of 3D graphics accelerators, a new direction of research, called Web Graphics, is emerging. This includes developing graphics applications as well as tools to support these applications in the Web environment.

March/April

Graphics Applications for Grid Computing

Grid computing allows access to distributed computing resources with the same ease as electrical power. In recent years, graphics application tools that take advantage of distributed computing, or grid environments, have emerged. New methodologies and techniques that harness resources for graphics applications are critical for the success of grid environments.

May/June

Advances in Computer Graphics

This issue covers an array of advances in computer graphics such as organic textures, lighting, and approximation of surfaces. Also, you'll find out about new developments in virtual reality, novel approaches in visualization, and innovative CG applications. The range of topics highlights the usefulness of computer graphics for everyone.

July/August

Nonphotorealistic Rendering

Nonphotorealistic rendering (NPR) investigates alternatives that leverage techniques developed over centuries by artists and illustrators to depict the world. One goal of this research is to broaden the achievable range of image styles and thereby embrace new applications. Additionally, NPR has the potential to open a new line of attack on one of the central problems of 3D computer graphics today: content creation.

September/October

Perceptual Multimodal Interfaces

This issue focuses on recent advances in methods, techniques, applications, and evaluations of multimodal interaction. Learn how researchers' cross-disciplinary approaches helped develop multimodal interfaces from interaction-centered prototypes to user-oriented and application-tailored solutions. This issue also explores the notion of moving toward transparent user interfaces.

November/December

3D Reconstruction and Visualization

Models based on 3D data will ultimately include everything associated with the environment, such as trees, shrubs, lampposts, sidewalks, streets, and so on. The main mode of exploration for this massive collection will be through interactive visualization. Ultimately, you should be able to fly continuously from overviews of a large city to centimeter-size details on the side of any building. Smoothly joining these different scales may require integrating rendering techniques in new ways.

http://computer.org/cga