

Markerless AR Tracking on a Mobile Device for Digital Art

Sun-Kyoo Hwang*, Ian Gwilt**, Mark Billinghurst*

* Human Interface Technology Laboratory New Zealand (HIT Lab NZ), University of Canterbury, Christchurch, New Zealand

** University of Technology, Sydney, Australia

ABSTRACT

In this paper we present a markerless digital art AR application working on a mobile device (PDA). In the installation a folder-shaped object is tracked and augmented with graphics. We have developed efficient vision algorithms to segment the folder object, find corner points and track the users view. The final PDA implementation runs at 6~7 frames per second and robustly tracks various camera poses to provide a compelling AR art experience.

KEYWORDS: Markerless AR, Mobile device, Digital art.

1 INTRODUCTION

Augmented reality (AR) requires real-time tracking of 3D camera pose for seamless augmentation of graphics. Black-square fiducial markers are commonly used for the fast and robust estimation of the camera pose [3][9].

Recently, AR has begun to be used in digital artworks [8], and in this case, the use of artificial markers may not be suitable. In addition, for a public exhibit it may be desirable for the AR application to be run on a hand-held device such as a PDA. Previous researchers have produced AR applications that run on mobile devices. However they used black-square marker tracking [9] or the performance was very slow without markers [7].

This paper presents a markerless AR system working on a PDA for a digital art installation. The AR application tracks a folder-shaped plastic object and overlays a virtual typographic graphic on the object. To get the system to work on a PDA, it was necessary to use efficient algorithms to find 3D camera pose. In order to speed up the computation we assumed that the background of the input image was simple. Our final tracking implementation worked at 6~7 Hz on a Dell Axim x51v PDA.

2 DIGITAL ARTWORK: 'SAVE_AS'

This augmented reality artwork builds on a body of work by the new media artist Ian Gwilt, who uses icons from the typical graphical user interface (GUI) as a starting point for his creative practice. In this artwork, entitled 'save_as', two upturned Perspex models of partially opened folders (220mm×180mm×80mm deep), can be seen attached to a gallery wall at about 1.5 meters off the ground (Figure 1). Visitors are free to walk up to the three-dimensional folders, which appear as enlarged sculptural representations of the typical folder icons that you would normally encounter on a computer desktop interface. A PDA with video camera attachment is supplied to the visitor, which allows the viewer to observe the virtual contents of the folders as if tumbling out towards the gallery floor.

This creative repositioning of the GUI plays with our understanding of the conventional computer interface, where computer icons no longer function as navigational devices and are

disenfranchised from the usual desktop metaphor through changes in context, scale and media type. The folders are overlaid with virtual texts that randomly link software command texts such as 'save', 'cut', and 'delete', with pronouns including 'him', 'her', 'them', to create word combinations like "save me", 'cut him' and 'delete her' (Figure 3). These word combinations question our relationship with these everyday technologies and activities.

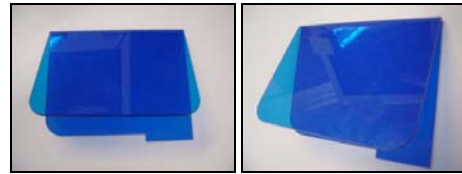


Figure 1. Folder objects which virtual graphics augmented

3 MARKERLESS AR TRACKING

A key part of this installation was using object based/markerless tracking to find the 3D camera pose from an input video image. The most important challenge was to develop efficient algorithms that can work on a low computing-power device such as a PDA. Our approach was to find four significant corner points of the folder object and compute the 3D camera pose by homography.

3.1 Labeling Based on Boundary Tracing

First, region labeling was performed to get the properties of each region. In an input image an interest object should be extracted for further processing. For the region labeling process the input color image was converted to a grayscale image and it is also converted to a binary image with a fixed threshold value.

From many choices we selected the labeling method of [2]. This was based on a combination of raster scanning and boundary tracing of objects within an image. The labeling method performs only one raster scan and does not require additional memory to record temporary labels. During the labeling process the coordinates of boundary pixels of each object can be recorded and are then used in the corner detection by curvature scale space. For the detailed algorithm of the labeling, refer to [2].

We computed the mean color, area size, and the length of boundary pixels of each object during the labeling process. This information was used to select the actual folder object. In the implementation we selected the one object which was bigger than a certain size with a mean color that was blue enough.

3.2 Corner Detection by Curvature Scale Space

In the previous process we extracted a feasible folder object by the labeling process. As we got the coordinates of the boundary of the folder object during the labeling process, we adopted the corner detection method based on the curvature scale space [1].

Curvature, κ , is defined as :

$$\kappa_{\sigma}(u) = \frac{\dot{X}_{\sigma}(u)\ddot{Y}_{\sigma}(u) - \ddot{X}_{\sigma}(u)\dot{Y}_{\sigma}(u)}{(\dot{X}_{\sigma}(u)^2 + \dot{Y}_{\sigma}(u)^2)^{1.5}}, \quad (1)$$

where $\dot{X}_\sigma(u) = x(u) \otimes \dot{g}_\sigma(u)$, $\ddot{X}_\sigma(u) = x(u) \otimes \ddot{g}_\sigma(u)$, $\dot{Y}_\sigma(u) = y(u) \otimes \dot{g}_\sigma(u)$, and $\ddot{Y}_\sigma(u) = y(u) \otimes \ddot{g}_\sigma(u)$. $g_\sigma(u)$ is a Gaussian function with the deviation σ and \otimes denotes the convolution operator. $\dot{g}_\sigma(u)$ and $\ddot{g}_\sigma(u)$ are the 1st and 2nd derivatives of $g_\sigma(u)$ respectively. In our implementation, we set $\sigma = 4$.

The corners were defined if the following was satisfied:

$$|\kappa_\sigma(u)| > T, \quad (2)$$

T is a threshold and adaptively determined. For details see [1].

3.3 3D Camera Pose Estimation from Homography

To find a 3D camera pose at least four correspondences of coplanar points are required [4]. In the previous step we found a set of corner points which are usually more than four. Therefore, we selected only four corner points to be used.

We assumed that the folder object in an input image was aligned almost in a fixed posture. The folder object consisted of two planar plates, one with a salient part which represented a folder name label. So, the upper two corner points in the folder object and the lower two corner points in the salient part were selected as the four feature points to compute 3D camera pose. We used the distance and angle information between two adjacent corner points and selected the four points by a heuristic method.

Figure 2 shows results from each step. Figure 2(a) is an input image. Figure 2(b) shows the result of region labeling where only the blue plastic object is extracted. In Figure 2(c) the corner points of the object are detected and displayed with red dot. Figure 2(d) shows four corner points selected from the points in Figure 2(c). The 3D camera pose is then calculated and a virtual cube is augmented on the object.

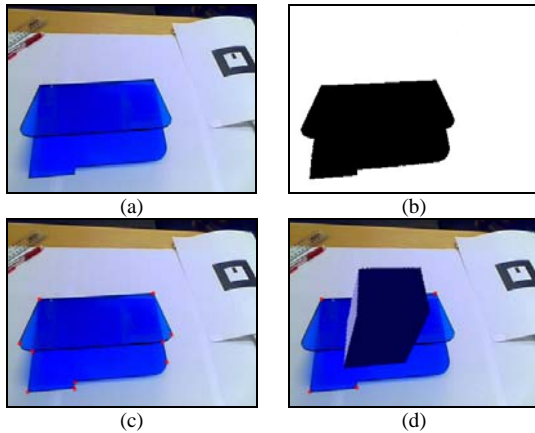


Figure 2. Computation of 3D camera pose from an input image.

4 RESULTS

In the implementation we used the Dell Axim x51v PDA with OpenGL ES [5]. As the PDA has an integrated multimedia accelerator (Intel 2700g), OpenGL ES can work very efficiently. The PowerVR SDK was used for the OpenGL ES implementation because it supports 2700g well [6].

A Spectec SD 300K camera was attached to the PDA to capture images. The camera provided 25~30 images per second when the image size was 320×240 pixels. We ignored the camera lens distortion to speed up the computation so no calibration is performed.

In order to make the system more robust it was important to remove the shadow effect by using a well-conditioned lighting configuration. In addition, the threshold value for binarization could be controlled by buttons on the PDA.

Table 1 shows the computational time for each operation on the PDA. Only 79 msec was taken for tracking the object and the rest of the time was spent for the input and output of an image. The system works at 6~7 frames per second.

Some working scenes are displayed in Figure 3. The graphical text was augmented exactly on the blue folder object at any 3D position of the camera. The AR installation was exhibited at the PowerHouse Museum in Sydney, Australia, from the 19th to 26th of August 2007.

Table 1. Computation time for each process

Operation	Times (msec)
Taking an image from a camera	34
Labeling	25
Corner detection	42
3D camera pose estimation	12
Display on the screen	37

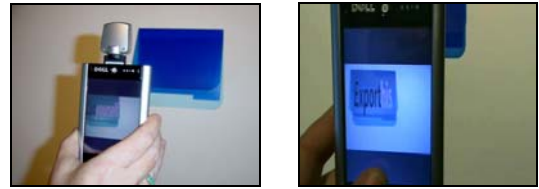


Figure 3. Markerless AR working on a PDA with detail (right).

5 CONCLUSION

In this paper we have presented a markerless augmented reality system working on a PDA as part of a digital artwork. Given a 3D rigid-body object we effectively extracted useful feature points and computed the 3D camera pose for seamless AR. The implementation shows that the proposed AR tracking works at 6~7 Hz on a Dell Axim x51v PDA without black-square marker. The technology we have developed will enable a wide range of handheld AR installations in the future.

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2006-352-D00150), and by the Australasian Centre for Interaction Design (ACID).

REFERENCES

- [1] X.C. He and N.H.C. Yung. Curvature Scale Space Corner Detector with Adaptive Threshold and Dynamic Region of Support. *International Conference on Pattern Recognition*, vol. 2, pages 791-794, August 2004.
- [2] Y. Ishiyama, C. Funaoka, and F. Kubo. Labeling Board Based on Boundary Tracking. *Pattern Recognition, 1992. Vol. IV. Conference D: Architectures for Vision and Pattern Recognition, Proceedings., 11th IAPR International Conference on*, pages 34-38, Sept. 1992
- [3] H. Kato and M. Billinghurst. Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, pages 85-94, 1999.
- [4] S. Malik. Robust Registration of Virtual Objects for Real-time Augmented Reality. *Masters Thesis, Carleton University*, 2002.
- [5] OpenGL ES, <http://www.khronos.org/opengles/>
- [6] PowerVR, <http://www.imgtec.com/PowerVR/Products/>
- [7] P. Riess and D. Stricker. AR on-demand: a practicable solution for augmented reality on low-end handheld devices. *AR/VR Workshop of the Germany Computer Science Society (Coblenz)*, Sept. 2006.
- [8] 'Save_as' artwork. <http://www.iangwilt.com/saveme.html>, Oct. 2007
- [9] D. Wagner and D. Schmalstieg. ARToolKitPlus for Pose Tracking on Mobile Devices. *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, February 2007