Video See-Through AR on Consumer Cell-Phones

Mathias Möhring, Christian Lessig, and Oliver Bimber
Bauhaus University
Bauhausstraße 11, 99423 Weimar, Germany,
{mathias.moehring, christian.lessig, oliver.bimber}@medien.uni-weimar.de

Abstract

We present a first running video see-through augmented reality system on a consumer cell-phone. It supports the detection and differentiation of different markers, and correct integration of rendered 3D graphics into the live video stream via a weak perspective projection camera model and an OpenGL rendering pipeline.

1. Motivation

To enable mobile devices, such as head-mounted displays and PDAs, to support video see-through augmented reality is a popular research topic. However, such technology is not widely-spread outside the research community today. It has been estimated that by the end of the year 2005 approximately 50% of all cell-phones will be equipped with digital cameras. Consequently, using cell-phones as platform for video see-through AR has the potential of addressing a brought group of end users and applications. Compared to high-end PDAs and HMDs together with personal computers, the implementation of video see-through AR on the current fun- and smart-phone generations is a challenging task: Ultra-low video-stream resolutions, little graphics and memory capabilities, as well as slow processors set technological limitations. We have realized a first prototype solution for video see-through AR on consumer cell-phones. It supports optical tracking of passive paper markers and the correct integration of 2D/3D graphics into the live video-stream at interactive rates. We aim at applications, such as interactive tour guiding for museums and tourism, as well as at mobile games.

2. Light-Weight Optical Tracking

The main objective of the implemented image analysis algorithm is to determine four non-coplanar points on a 3D paper marker. They are used as input parameters for rendering the overlaid 3D geometry correctly using a weak perspective projection to model the camera-to-image plane transform (see section 3). Using such a concept instead of determining the extrinsic parameters of the phone's camera relative to the marker ensures that our approach does not depend on the camera's intrinsic parameters. Thus it can be applied in combination with any camera without having to correct lens distortion and measuring focal lengths. In addition, the algorithm has to recognize individual markers by reading a circular barcode. Using colored features, such as lines and circles on the marker allows searching for increases in RGB channels instead of having to detect a specific color or greyscale. This makes the tracking process robust and – in combination with a specific spatial distribution of the features on the marker – allows encoding a large number of different codes.

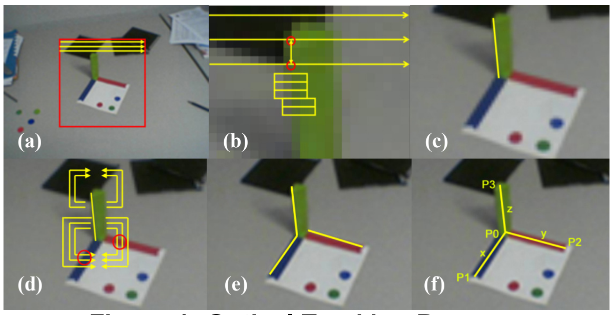

Figure 1: Optical Tracking Process.

To achieve interactive frame rates, a multi-level scanning algorithm has been developed: Each frame is scanned with continuous, parallel scan lines in every n-th pixel-row or column (fig.1-a) until an expected color increase has been detected in two consecutive scan lines. The slope of the line that connects the two feature intersections is computed next (fig.1-b). It gives a first estimate of the gradient of the line-feature's edge. The edge is traced in both directions using a one-dimensional search window that is normal to the gradient until the two endpoints are reached. The result is the edge and the endpoints of the most vertical marker axis in the image (fig.1-c). The other two axes have to emerge from one of the detected endpoints. To find them, we apply scan tracks that orbit the endpoints in rectangular paths with increasing radii until two consecutive intersections on two different line features are found (fig.1-d). This can only be the case for one endpoint. As for the first axis, the slopes between the intersections are computed to estimate the gradients of the edges which are used to trace them in both directions until their endpoints are reached. The results are the edges of the remaining axes and their four endpoints (fig.1-e). During the edges are traced, their average width is determined and continuously updated. This allows to estimate the center of the axes and to correct the position of the three line features (fig.1-f). Finally, the three endpoints which intersect are merged into a single one. The remaining four endpoints of the line features represent the basis points of the marker coordinate system. The result of this is the projections of three endpoints and one origin point that are non-coplanar on the 3D marker. These points span rectangular reference frames in the image that are sampled for the projections of circular features on the marker using a simple bilinear-linear interpolation. Spatial and color appearance of circular features lead to a predefined barcode. Furthermore, interframe information is used to narrow the search region and to eliminate impossible results (fig.1-a).

3. Perspective Rendering

To render the overlaid 3D graphics in the correct perspective relative to the cell-phone and the marker, the geometry has to be transformed into a global affine coordinate frame. This coordinate frame is defined by the four non–coplanar basis points determined during the optical tracking step. The pixel positions of vertices can be approximated by projecting them with respect to the detected pixel positions of the four basis points. This approximation can be fully integrated into the common OpenGL rendering pipeline (like in OpenGL ES). Thereby the first two rows of the projection matrix are filled with vectors, calculated from the u and v pixel coordinates of the basis points, affecting x and y coordinates of each vertex and simultaneously spanning the virtual camera's plane in the affine coordinate system. The third row is defined by the cross product of the other two vectors. This vector, however, does not provide correct depth information, but provides the correct depth ratio between all vertices. To use this matrix in the OpenGL transformation pipeline, an additional scaling has to transform the resulting values into normalized device coordinates and the correct depth range. While the camera image is displayed in the background, 3D graphics is overlaid by rendering it into the affine coordinate system using the customized OpenGL pipeline. The lack of a floating point unit on consumer cell-phones requires a fast fixed point arithmetic that avoids divisions.

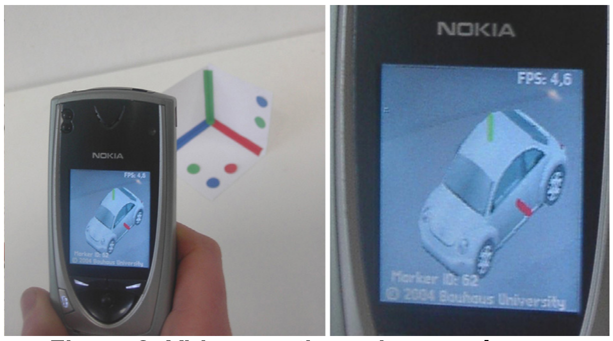

Figure 2: Video see-through example on a consumer cell-phone.

4. Results

The choice of a weak perspective projection camera model and an affine object representation does not require to know the camera's intrinsic and extrinsic parameters in terms of rendering three-dimensional graphics perspectively correct into the video stream. This approach can deal with the high optical distortion and the low image resolution without requiring the end user to perform any camera calibration step.

Our phone's frame grabber delivers 16 images per second with a resolution of 160x120 pixels and a color depth of 12 bits. Detecting the marker and reading the barcode requires approximately 25% performance - thus we achieve an average frame rate of about 12fps. The performance of the rendering step depends on the complexity of the scene. In the example shown in figure 2, the Beetle consists of 404 triangles, 6 textures and was Gouraud-shaded with a single point light source. The final frame rate which includes frame grabbing, marker and barcode detection, as well as rendering is 4-5fps. The marker size is 10x10x10cm and can be detected in a range of 30cm – 1.5m.

Acknowledgements

A pre-release version of OpenGL ES / NGL and technical support were provided for this project by Nokia Research.

References

[1] Vallino, J., Kutulakos, K.N. 2001. Augmented reality using affine object representations. Fundamentals of Wearable Computers and Augmented Reality, pp. 157 – 182, ISBN: 0-8058-2902-4, Lawrence Erlbaum (publisher).