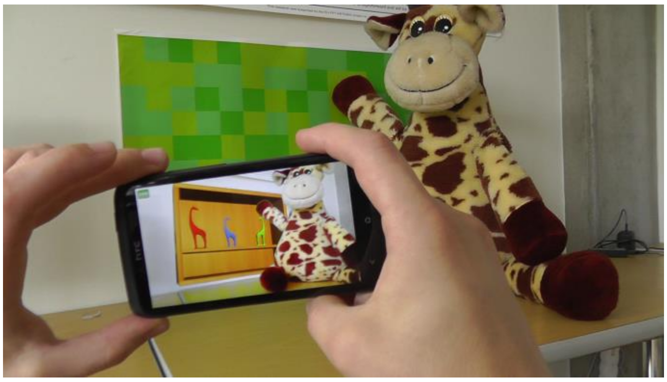
3D Pose Estimation on Android Using Uniform Marker Fields

V3C

Chroma-keying is the most widespread method for augmenting real-world scenes with virtual elements by replacing marked real-world objects. The process typically requires considerable manual intervention. High-precision movement coherence between real and virtual objects is achieved by complicated and heavy camera grid. This motivated towards providing a lightweight solution for automatic camera pose estimation and calibration. Our goal was to provide a solution that works also on common mobile phones. We achieved this by using a low-contrast Uniform Marker Fields marker in the shades of green (blue) inserted into the scene. This solution improves the flexibility of the system and provides real-time preview of the augmented scene.

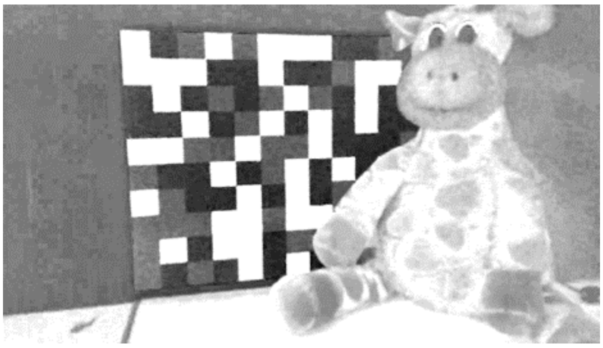


The figures show the green-screen low-contrast Uniform Marker Field (UMF) in the background. The scene is viewed by a mobile phone. The green-screen is filtered out using chroma-keying and replaced by a shelf with 3D giraffes. The application runs on any mid-range Android phone.

Our research

We implemented the 3D Pose Estimation on Android Using Uniform Marker Fields for chroma-keying in the form of a Unity 3D plugin. Unity 3D is a free and cross-platform 3D game engine. It supports both of our targeted platform – PC and mobile ARM (Android).

The full algorithm has three main parts: chroma-keying process, tracking&detection and camera pose estimation. We optimized the performance of the first two main parts. We used OpenGL shaders and the Halide language to generate accelerated implementations using the NEON instruction set.



The plugin receives the input on the ARM platform directly from the underlying Android operating system. For the chroma-keying process the input data is converted into YUV color space. The algorithm then computes the foreground mask and a grayscale map (see image above), which used further for camera pose estimation. The whole process is accelerated for high-quality viewing using the graphics chip by OpenGL ES shaders. For the UMF detector we used a low-quality simplified chroma-keying accelerated using NEON instructions. We also accelerated gradient computations, sub-sampling and Gaussian blur.

Results

For the performance evaluations we used VGA resolution. We tested our solution both on PC (Intel Core i7 2.2GHz) and ARM platforms (ARMv7 1.5 GHz). The Android implementation was running at 33.7 FPS on the foreground, and was processing frames asynchronously for the camera pose estimation in 23.4 ms including Java overhead. The breakdown of the measured computational times are shown in the table below. The fractions of time reported do not add up due to the system overhead. All times are in milliseconds.

Platform [ms]	**Total**	(chroma-keying	Tracking and detection	Camera pose)
PC	**10.7**	3.7	1.9	1.4
PC with Halide	**6.3**	0.2	1.9	1.4
ARM	**25.3**	4.8	14.0	2.1
ARM with Halide	**23.4**	4.1	12.8	2.1

Our experiments show, that our solution is efficient and is able to run smoothly on mid-range mobile phones and tablets. The functional software can be used as a cheap and low-maintenance simulcam. Also, the solution will allow for live storyboarding and virtual shot prototyping, which are only dreamt-of at the current state of the art.

Title: 3D Pose Estimation on Android Using UMF
Author: István Szentandrási (iszent@fit.vutbr.cz)

Faculty of Information Technology
Brno University of Technology
Božetěchova 1/2
612 66 Brno, Czech Republic
Tel.: +420 54114-1144
E-Mail: info@fit.vutbr.cz

Research Project: V3C - Visual Computing Competence Center
No. TE01020415
www.v3c.cz
Project Manager: Pavel Zemčík (zemcik@fit.vutbr.cz)
Tel.: +420 603 487 824

Technologická agentura České republiky

Centra kompetence

Acknowledgment

This work was supported by the Technology Agency of the Czech Republic.