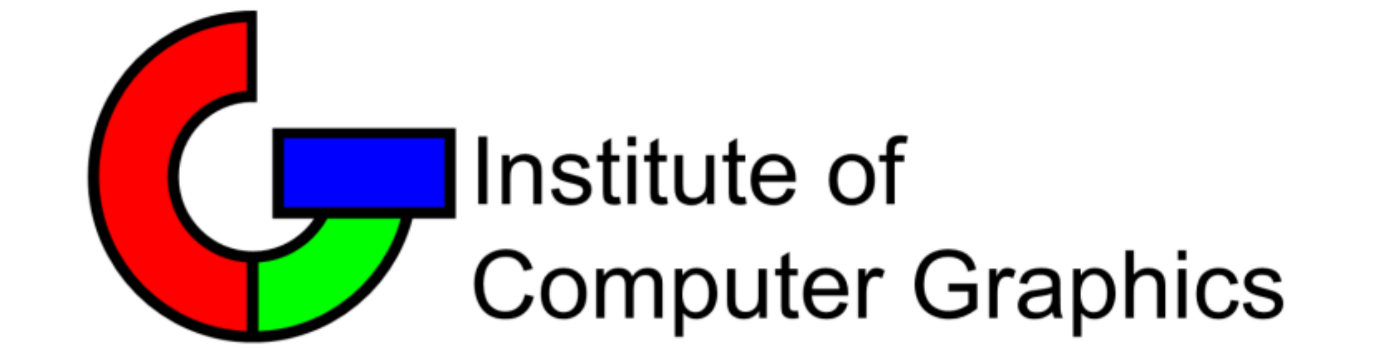


# Display Pixel Caching

Clemens Birkbauer<sup>1</sup>, Max Grosse<sup>2</sup>, Anselm Grundhöfer<sup>2</sup>, Tianlun Liu<sup>1</sup>, and Oliver Bimber<sup>1</sup>  
<sup>1</sup> Johannes Kepler University Linz, <sup>2</sup> Bauhaus-University Weimar



## Abstract

### Introduction and Motivation

A variety of standard video modes that stretch or crop lower resolution video content linearly to take full advantage of large screen sizes have been implemented in TV sets. When content and screen aspect ratios differ, format proportions may be compromised, video content may be clipped, or screen regions may remain unused. Newer techniques, such as video retargeting and video upsampling, rescale individual video frames and can potentially match them to the display resolution and aspect ratio. However, none of these methods can display simultaneously more than is contained in a single frame.

### Display Pixel Caching

We present a new video mode for television sets that we refer to as display pixel caching (DPC). It fills empty borders with spatially and temporally consistent information while preserving the original video format. Unlike related video modes, such as stretching, cropping, and video retargeting, DPC does not scale or stretch individual frames. Instead, it merges the motion information from many subsequent frames to generate screen-filling panoramas in a consistent manner (cf. figure 1). In contrast to state-of-the-art video mosaicing, DPC achieves real-time rates for high-resolution video content while processing more complex motion patterns fully automatically.

## Related Work

### Video Retargeting

Video retargeting approaches, such as [1,2], apply a non-linear rescaling of input footage to fit it into other format ratios. Techniques that use interactive constraint editing (e.g., [1]) or global processing of the entire video cube (e.g., [2]) achieve good quality but avoid online processing. Methods that do support online processing achieve near real-time rates (e.g., 10fps for 720p [1]), but at moderate quality and robustness.

### Video Upsampling

Video upsampling, such as in [3], increases the spatial resolution of videos and is real-time capable, but does not adapt the aspect ratio and - as video retargeting - displays only the content that is present in individual frames. However, it can be applied as a preceding step to any other video mode (including DPC).

### Video Completion

Video completion and motion inpainting methods can estimate motion in clipped video frame regions to guide a filling process that considers corresponding pixels in neighboring frames. They can, for example, support video stabilization [4] and are presently not real-time capable (e.g., 2fps for 720x486 in [4]). In contrast to common video completion and inpainting, DPC fills much larger frame regions in real time.

An application of video completion to video extrapolation for filling large border regions in video frames iteratively was presented in [5]. To avoid visual artifacts, this method removes salient content and applies blur that increases towards the outer edges of the extrapolated regions. Therefore, these regions are only suitable for peripheral vision and are only applicable for displays that cover a large field of view and restrict eye movement, such as head-mounted displays. This method is also not real-time capable.

### Video Mosaicing

Video mosaicing techniques generate a mosaic image from a frame sequence by registering and blending the recorded video footage. Generally, such mosaicing techniques are optimized for both static scenes and controlled camera movements, such as pans, tilts, and translations. They can achieve real-time rates for moderate resolutions (e.g., 15fps for 640x480 in [6]). For unstructured camera motions, for arbitrary scenes that cause parallax effects, and for complex mixtures of global camera motion and local object motion, the motion flow in the images can vary substantially in different regions. While conventional video mosaicing fails in such a situation, this is the typical use case for DPC. Techniques that apply more advanced alignment and blending strategies (e.g., [7,8]) reduce local misalignments but are too expensive for real-time applications at high resolutions.

## Processing Pipeline

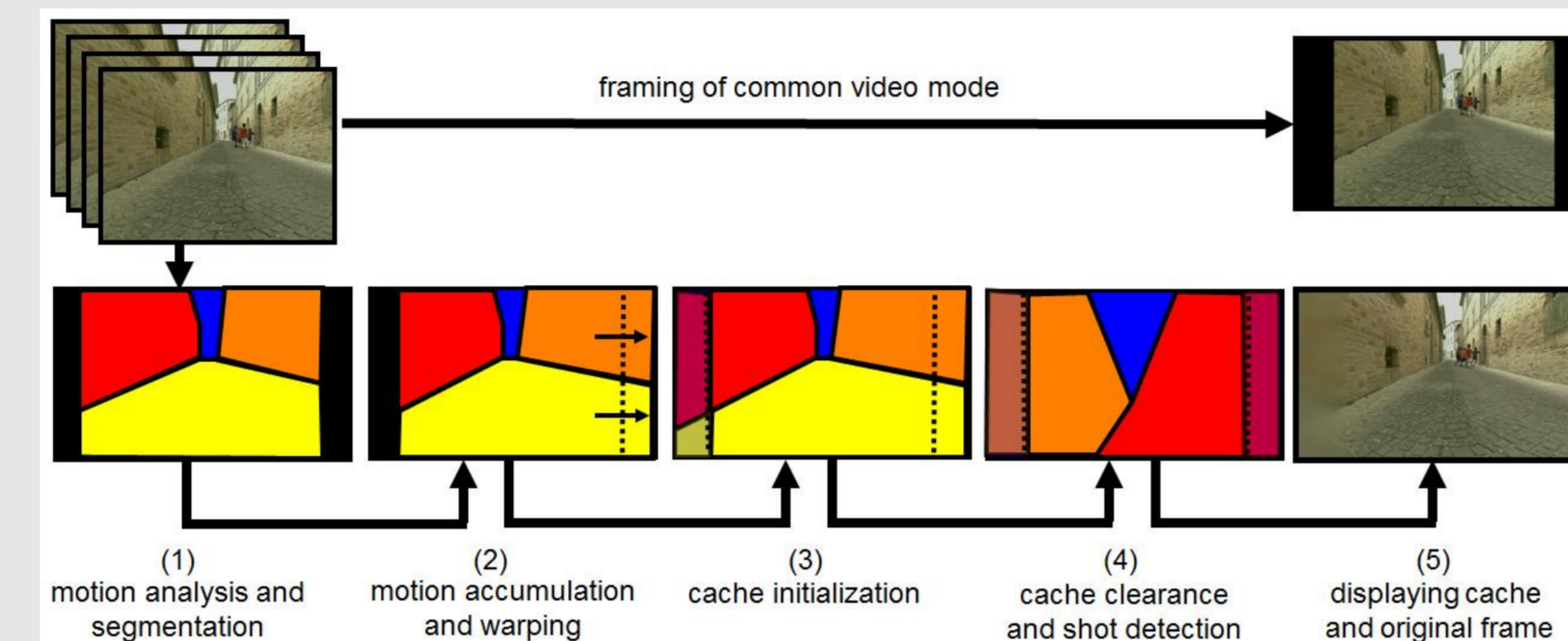


Figure 1: Overview over DPC processing steps.

## Cache Creation

### Motion Analysis and Segmentation

To identify different motion layers, we process the block-wise motion flow field provided for P-frames of MPEG-encoded input streams. For segmenting the flow field into different motion layers, we apply a conditioned RANSAC multiple times. Each layer is represented by a 3x3 homography matrix.

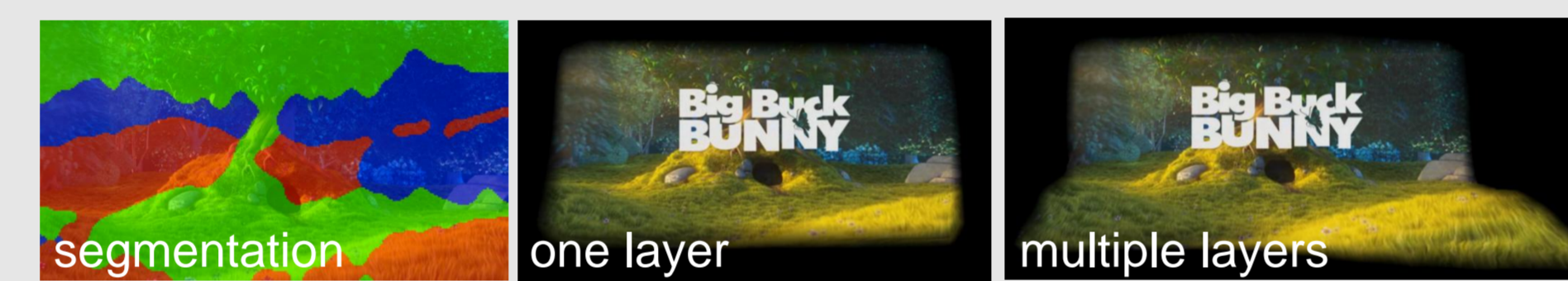


Figure 2: Multiple frames of a forward motion warped incorrectly with a single layer, and same frames registered correctly with multiple layers per frame.

### Motion Accumulation and Warping

The homographies of the determined motion layers can now subsequently be used for warping previous frames (cf. figure 3). We store the current frame and preceding frames together with their homography assignments in a history ring-buffer.

For warping we initially define a regular 2D vertex grid in the resolution that equals the motion-vector resolution of the MPEG data. Then we traverse the history ring-buffer in front-to-back order. For each step, we texture-map the frame onto the current instance of the grid and then render and alpha-blend the result into a composite image. Next, we warp the grid by multiplying each grid point with its corresponding homography matrix and continue to the next older frame.

When the maximum size of the ring-buffer is reached, the oldest frame is moved from the history ring-buffer to an accumulation buffer. This buffer is transformed in a similar way as the frames in the ring-buffer.

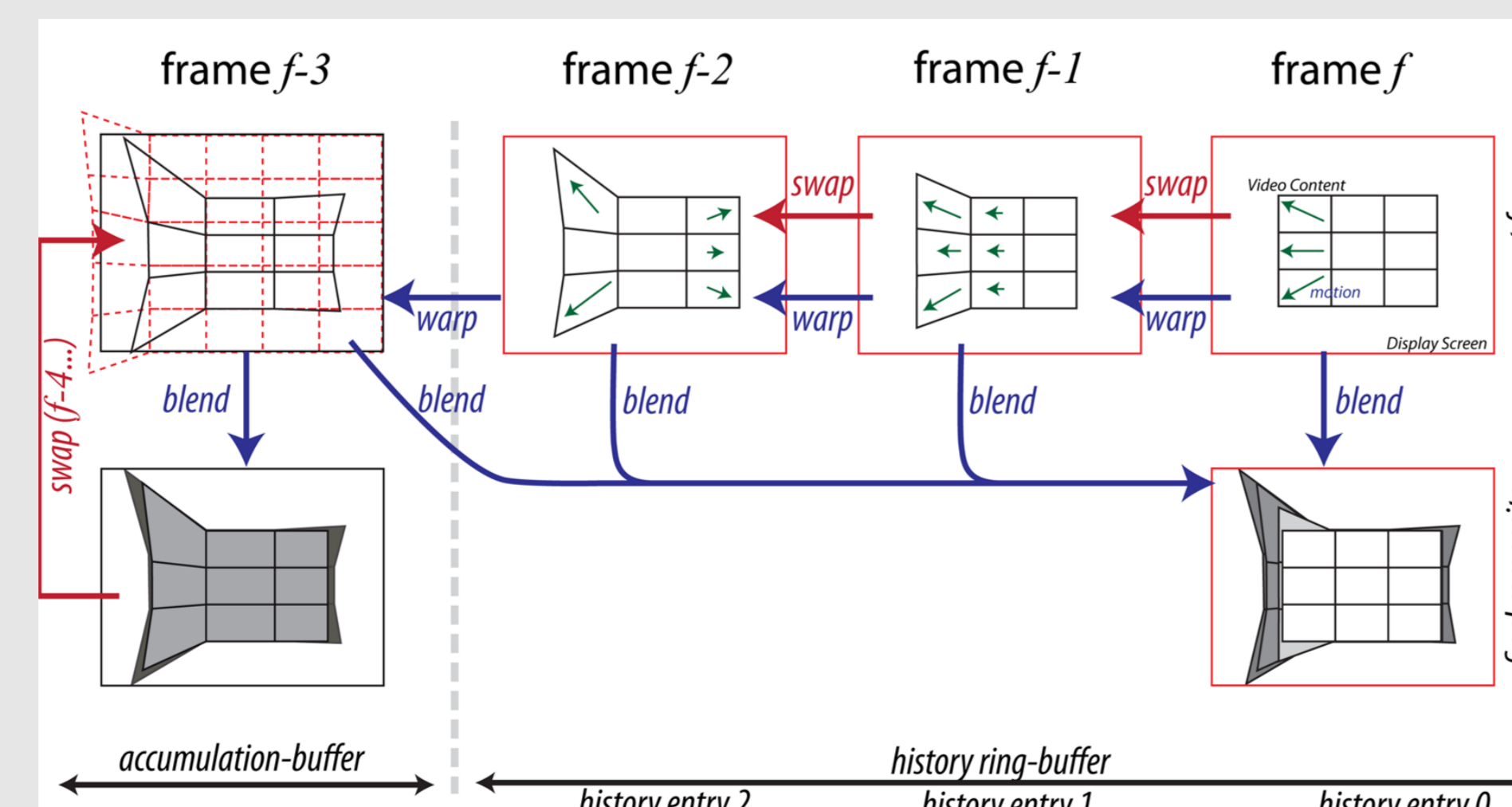


Figure 3: Image warping with ring buffer and accumulation buffer.

## Cache Handling

### Cache Initialization

The border edges in the cache are generally neither straight nor screen-aligned (cf. figure 4a), which can be disturbing for viewers. Alternatively we can initialize unfilled portions of the cache with a smooth color transition (cf. figure 4b). To reduce the blurred areas created by a full initialization, we can optionally clip it to screen aligned edges. One possibility is to place the clipping edges progressively at the outermost extent of the original cache content (cf. figure 4c). Another possibility is to place the clipping edges conservatively at the inner limit of the original cache content (cf. figure 4d).

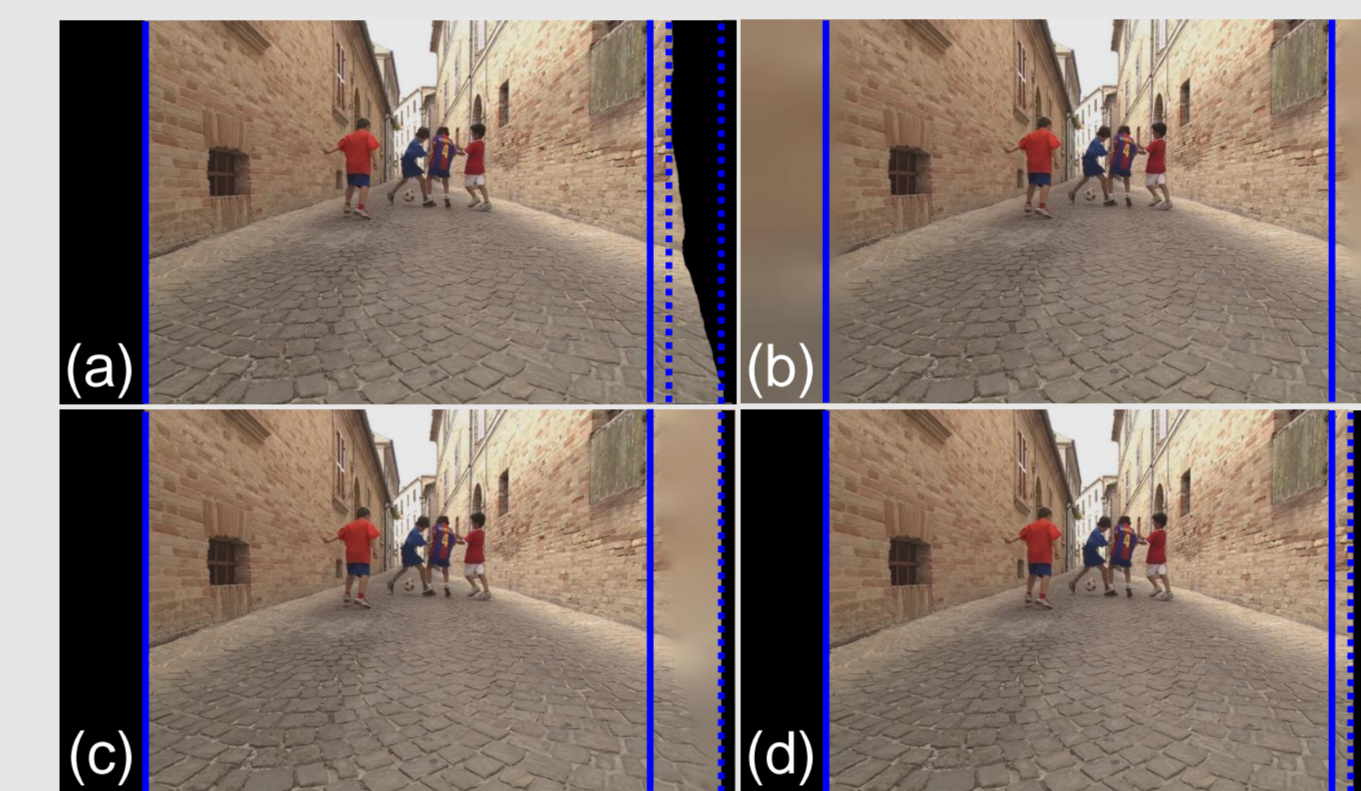


Figure 4: Cache initialization options.

### Cache Clearance and Shot Detection

We detect hard cuts and fading transitions in the video by analyzing the information transport between the video frames. The cache content is cleared or faded in the same way as the original video content.

## Results

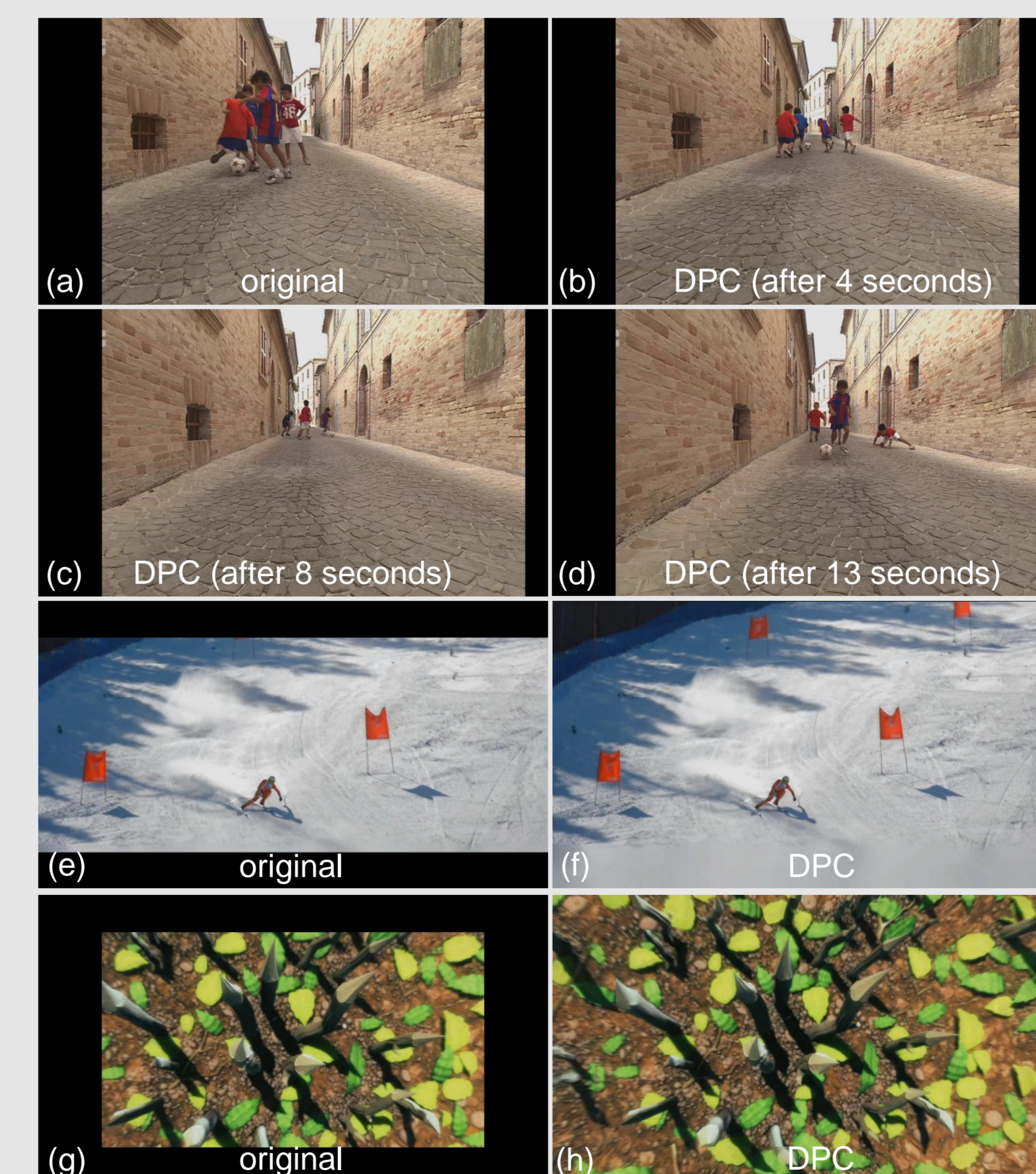


Figure 5: Results of DPC with leftward camera motion (a-d), downward motion and pre-filled cache (bottom border) (e,f), forward motion (g,h).

## Evaluation

### Performance

To achieve adequate load balancing, we decoded the videos via FFmpeg on the CPU while executing the DPC video processing pipeline in parallel on the GPU using CUDA and OpenGL shaders.

DPC achieves real-time rates for high-resolution video content (e.g., 50fps for PAL videos displayed on a 720p screen, 29fps for PAL when displayed on a 1080p screen, and 26fps for 720p displayed on a 1080p screen) while processing complex motion patterns fully automatically.

### User Evaluation

To determine how DPC performs compared to the alternative display modes, and to identify the optimal cache initialization option, we carried out a user evaluation. Figure 6 shows the averaged results of our 59 participants.

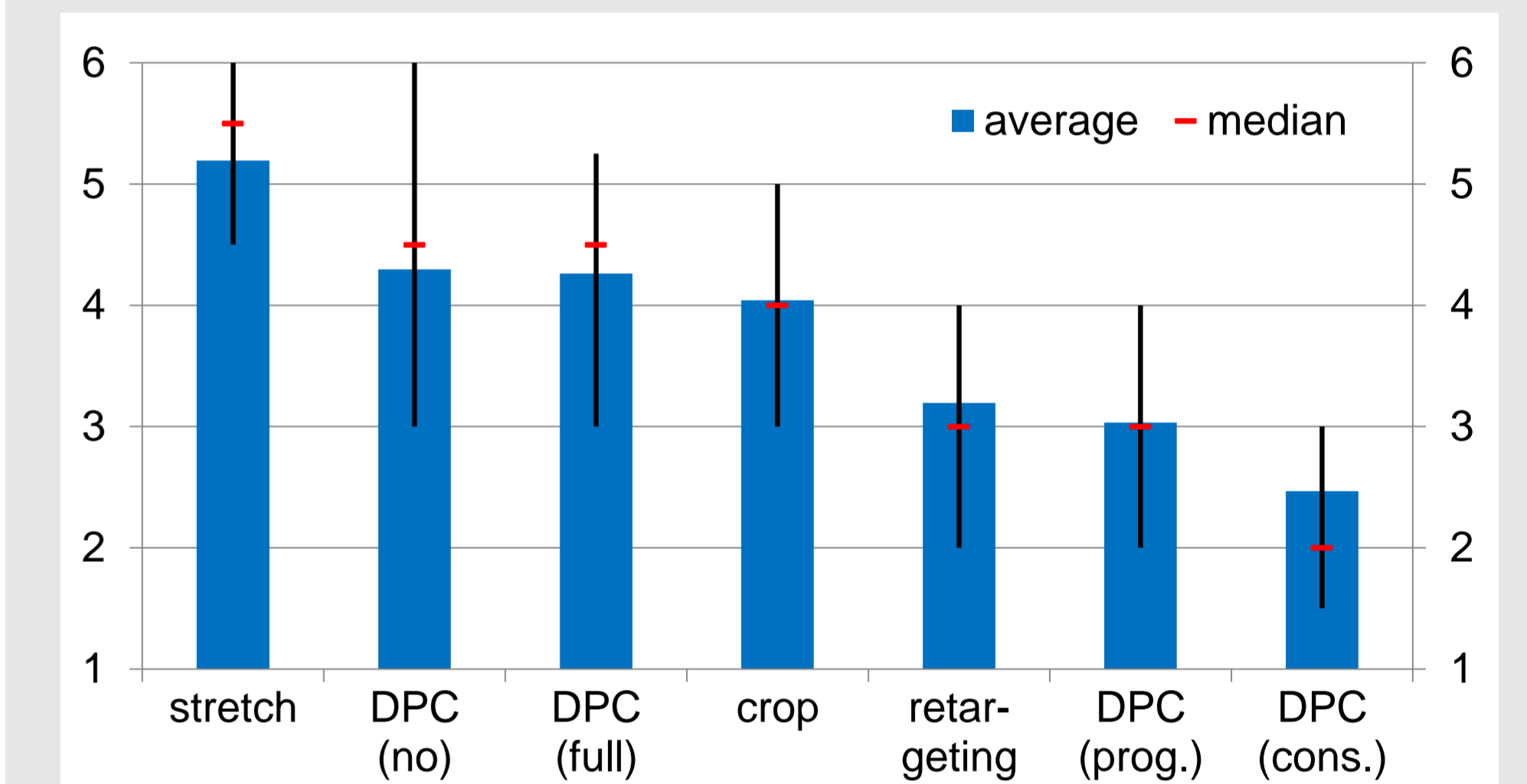


Figure 6: Results of the user evaluation.

We presented side-by-side pairs of video sequences in their original (unmodified) format and with one of the following modes: stretch, crop, retargeting, and DPC with the four cache initialization options. For each of the seven modes, two video samples were shown, one in 4:3 and one in cinemascope format, displayed as on a 16:9 screen. The subjects were asked to compare each video mode with the original format and to indicate their preferences by scoring (1 = mode preferred most, 6 = original preferred most).

## References

1. Krähenbühl, P., Lang, M., Hornung, A., Gross, M.: A system for retargeting of streaming video. In: SIGGRAPH Asia'09: ACM SIGGRAPH Asia 2009 papers, ACM (2009) 1-10
2. Wang, Y.S., Lin, H.C., Sorkine, O., Lee, T.Y.: Motion-based video retargeting with optimized crop-and-warp. In: SIGGRAPH'10: ACM SIGGRAPH 2010 papers, New York, NY, USA, ACM (2010) 1-9
3. Freedman, G., Fattal, R.: Real-time gpu-based video upscaling from local self examples. In: ACM SIGGRAPH 2010 Talks. (2010)
4. Matsushita, Y., Ofek, E., Ge, W., Tang, X., Shum, H.Y.: Full-frame video stabilization with motion inpainting. IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (2006) 1150-1163
5. Avraham, T., Schechner, Y.: Ultrawide foveated video extrapolation. IEEE Selected Topics in Signal Processing 5 (2011) 321-334
6. DiVerdi, S., Wither, J., Höllerer, T.: Envisor: Online environment map construction for mixed reality. In: IEEE Virtual Reality 2008. (2008) 19-26
7. Shum, H.Y., Szeliski, R.: Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment. Int. J. Comput. Vision 36 (2000) 101-130
8. Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. International Journal of Computer Vision 74 (2007) 59-73

<sup>1</sup> {firstname.lastname}@jku.at  
<sup>2</sup> {firstname.lastname}@uni-weimar.de