Unstructured Video-Based Rendering: Interactive Exploration of Casually Captured Videos

Luca Ballan, ETH Zürich
Gabriel J. Brostow, University College London
Jens Puwein, ETH Zürich
Marc Pollefeys, ETH Zürich

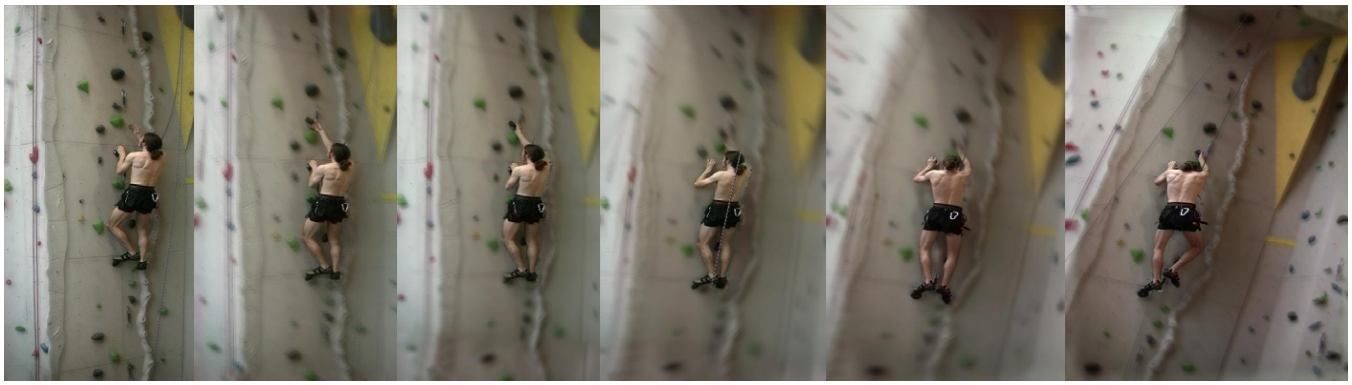

Figure 1: *Navigating multiple videos of a climber. First and last images are from real cameras 40° apart.*

Abstract

We present an algorithm designed for navigating around a performance that was filmed as a "casual" multi-view video collection: real-world footage captured on hand held cameras by a few audience members. The objective is to easily navigate in 3D, generating a video-based rendering (VBR) of a performance filmed with widely separated cameras. Casually filmed events are especially challenging because they yield footage with complicated backgrounds and camera motion. Such challenging conditions preclude the use of most algorithms that depend on correlation-based stereo or 3D shape-from-silhouettes.

Our algorithm builds on the concepts developed for the exploration of photo-collections of empty scenes. Interactive performer-specific view-interpolation is now possible through innovations in interactive rendering and offline-matting relating to i) modeling the foreground subject as video-sprites on billboards, ii) modeling the background geometry with adaptive view-dependent textures, and iii) view interpolation that follows a performer. The billboards are embedded in a simple but realistic reconstruction of the environment. The reconstructed environment provides very effective visual cues for spatial navigation as the user transitions between viewpoints. The prototype is tested on footage from several challenging events, and demonstrates the editorial utility of the whole system and the particular value of our new inter-billboard optimization.

1 Introduction

Photo- and video-collections exist online with copious amounts of footage. Community-contributed photos of scenery can already be registered together offline, allowing for navigation of specific landmarks using a fast Image-Based Rendering (IBR) representation [Snavely et al. 2006]. We propose that similar capabilities should exist for video of performances or chance events, filmed by multiple passersby. The BBC, Reuters, and many other organizations already collect and use video from Citizen-journalists. This allows them to better cover important events such as crimes, catastrophes, and performances. We expect that the coverage of events by casually captured videos will increase. Organizing these videos is difficult, because there are many potential ways a user or production editor may want to navigate them, and just playing the multiple videos in series or jump cutting between them may not convey the important motion and context of the event.

The aim is to give the user the ability to replay an event by navigating around a performer, using footage from a multi-view collection of videos. One can only make weak assumptions about the footage because the audience members doing the filming may have various video-recording devices, they could sit or move about far apart from each other, indoors or out, and they may have a partially obstructed view of the action. Unlike photos of a single place that sample space with boundless density [Schindler and Dellaert 2010], videos of an event must coincide in space and time. We must be realistic about the actors and the types of videos available in such situations, because that dictates the possible nature of the navigation. For this reason, it is safest for the new video-based renderer to adhere to actual recorded video when possible, or to interpolate a view of the action only along virtual paths between real cameras.

Our work shares many challenges with but is distinct from variants of Free Viewpoint Video such as [Kilner et al. 2006] and [Carranza et al. 2003], which seek a full 3D polygonal reconstruction of the actors. We are intentionally relinquishing the ability to move around freely to accommodate i) a greater variety of acceptable input footage (viewing angles and subjects), and ii) a closer adherence to the look and feel of the actual videos. Also, unlike the freeze-motion effects used in "The Matrix" and for inspecting crit-

ical moments in football and soccer, we want the option of having action continue while we navigate among the videos. We believe that in many situations, it is more valuable to see the motion in progress than to freeze time and inspect just one instant. Freezing time is in some ways harder since greater scrutiny of details is possible from all sides, but also easier because the situation reduces to one of multi-view static object reconstruction. For such situations outdoors where custom scene models exist, we defer to the formidable accomplishments of [Guillemaut et al. 2009; Guillemaut et al. 2007], [Kanade 2001], and [Würmlin and Niederberger 2010], which have been used for sports broadcasting purposes.

The main contributions of this work are i) the very ability to interact with and navigate casual footage of performances for the first time, and ii) the perceptual smoothness of the inter-camera transitions. To reach those goals, we developed an interactive algorithm to synthesize a hybrid representation with textured surface geometry for the scenery, and billboard-based rendering of the actor. The segmentation of the moving performer is significantly refined by using our own renderer to build a 3D variant of Video Matting [Chuang et al. 2002] (Section 3.3). The inter-billboard distance measure (Section 4.2) and performer-specific view interpolation (Section 4.1) are key to navigating between cameras whose angular separation is rarely less than 30°.

1.1 System Overview

The proposed algorithm can be separated into three stages. The user has varying degrees of control over each stage. It is assumed that the multiple videos have already been collected and identified as featuring the same events.

Scenery & Offline Processing The input data must first be processed to synthesize the hybrid representation that will be navigated. Each video recording device has its own clock and framerate. Like [Hasler et al. 2009], one can use correlation of the audio signals to synchronize the video streams automatically. We silence the quieter 90% of each video, align on the rest, and still need to manually timeshift about one in four videos. Sound travels slowly, so video-only synchronization [Sinha and Pollefeys 2004; Tuytelaars and Van Gool 2004] may be preferable despite being more costly computationally.

The system processes videos of the event and any available photographs of the area to reconstruct a 3D surface mesh and view-independent texture maps. We use the robust dense geometry reconstruction technique of [Zach et al. 2007]. Online 3D models of earth are improving in coverage and accuracy, to the point that some algorithms take such geometry for granted [Kopf et al. 2008]. Our system can import parts of such scene models also. The video cameras' 3D poses are computed relative to the scene geometry. As a final preprocessing step, the user selects-by-painting on the performer of interest in a few frames of each camera's video. This markup serves two purposes, namely to indicate which performer should stay in focus during the Online Navigation phase, and to learn a color model which will be used to automatically compute video mattes for the cameras.

Online Navigation The user navigates in either Regular Mode or Orbit Mode with a live preview, playing the timeline forward or reverse, and gliding between neighboring camera's vantage points. The motion of the virtual camera is automatically adapted to account for the changing locations of the designated performer and the real cameras. An optimization chooses parameters for the transition from one camera to the next, to better conceal visual artifacts inherent in VBR of casually captured videos. The user can make artistic choices among rendering styles for the performer's context.

Offline Postprocessing Optionally, after the online navigation is recorded, the system can automatically perform extra computations to re-render a version at higher-quality. Improvements over the online rendering include use of the highest resolution images, performing a deeper search of possible transition parameters, gradual color correction transition effects, and simple audio transitions. Presented results are without postprocessing unless specified.

2 Background

This approach to navigating the footage of a performance benefits from a variety of developments in user-interaction, graphics, and vision. VBR of real footage, just like IBR [Chen and Williams 1993], depends on the density of available footage. In the context of photo inpainting at least, [Hays and Efros 2007] showed that some applications only become viable with immense quantities of footage. We demonstrate that even before online video collections reach such a critical mass, interesting interactive video navigation is possible through our proposed technique. The related work can be grouped into the following four areas.

Photo & Video Navigation Digital photos and videos can be organized and explored with various applications in mind. Individual photos are geo-tagged and anchored to pop up when one zooms to that part of the globe in Google Earth or Microsoft's Virtual Earth. Local collections of normal photos can be browsed and edited using Adobe Bridge for example, while community collections are organized in online sites, often sorted based on user-annotations. Research in this area has culminated in the Photo Tourism work of [Snavely et al. 2006; Snavely et al. 2008] and the commercially supported online PhotoSynth community. One of their main contributions was the pivotal insight that instead of stitching many people's disparate photos together into a panorama, it is possible and useful to compute a 3D point-cloud from the 2D features that the photos have in common. The point-cloud in turn serves as a scaffold and a non-photorealistic backdrop that provides a spatial context. While a "visitor" navigates the original photos, they see the point-cloud and hints of other photos in a way that reflects the real spatial layout of, for example, the Trevi Fountain.

In-between views are generated during the transition from one photo to the next. [Snavely et al. 2006] use a planar morph to cross-fade the two photos. They found that this usually made fewer visual artifacts than a triangulated mesh morph, despite being less faithful to the non-planar geometry in the scene. Their subsequent work [Snavely et al. 2008] gave one the option of specifying an orbit-point. Forcing the planar proxy to go through the orbit-point has the effect of stabilizing the user-selected part of the scene as the virtual camera orbits between photos with small angular deviations. The billboard part of our hybrid representation is positioned in space similarly, but serves as proxy geometry for the moving foreground actor, rather than the background scene.

Navigation and interaction with videos has new challenges and opportunities. [Sivic and Zisserman 2003] showed the viability of example-based search for objects or people appearing in full-length movies. For our purposes, one can imagine eventually using such research to scour the internet for more multi-view footage of a particular event. An interesting new paradigm for more interactive video-playback has emerged in parallel from several research groups [Karrer et al. 2008; Dragicevic et al. 2008; Goldman et al. 2008]. By offline clustering of the optical flow vectors throughout a video, the user can then play back only "relevant" frames. For example, by clicking in the area of a car and dragging the mouse

along a path, one sees the frames when the car finally did drive that way. These algorithms operate over time in a 2D space, but [Goldman 2007]'s version already explored a fascinating variety of production-relevant applications.

Narrow Baseline View Interpolation IBR had normally been developed for high quality realistic representations of static scenes ([Levoy and Hanrahan 1996] and [Gortler et al. 1996]. The special cases, where a realistic 3D proxy object is available, have slightly relaxed filming-angle requirements [Buehler et al. 2001; Heigl et al. 1999]. [Waschbüsch et al. 2007] show good free-viewpoint renderings when footage is acquired in a studio with multiple structured-light 3D capture systems, and stationary high quality cameras. Like us, part of their pipeline uses billboards, but theirs are substantially enhanced with available 3D information. [Zitnick et al. 2004] found that even sequences with highly dynamic human motion could have temporally coherent per-pixel depth estimates of sufficient accuracy to allow stunning view interpolation between camera pairs. They spanned a total of 30° of viewing angle using a chain of eight cameras, and could tolerate 100 pixels disparity by focusing special computations on depth discontinuities. In situations where the inter-camera baseline is small and the scene controlled, this appears to be the best system for view interpolation of motion. The recent work of [Stich et al. 2008] demonstrates that under conditions of even 15° angular separation, it can be sufficient to model the whole scene with homography transformations of 2D superpixels whose correspondence, based on sparse feature matches, is computed as an alternative to per-pixel optical flow. The demonstrated examples are impressive, with revealing errors naturally occurring in the largely low-texture areas. Also in a studio setting but starting with crude geometry of a performer, [Eisemann et al. 2008] show how good optical flow can fix texture-assignment problems that occur where views of some geometry overlap. Previously, view interpolations based on epipolar constraints was demonstrated by [Seitz and Dyer 1996], where correspondences were specified manually. However normally, these view interpolation algorithms rely heavily on correlation-based stereo and nearby cameras.

Visual Hull Techniques The opposite case of widely separated cameras lends itself to shape-from-silhouette techniques. Since excellent figure/ground image segmentation is possible in studio conditions, there is an established thread of research focused on converting multi-view silhouettes into visual hulls [Matusik et al. 2000; Franco and Boyer 2005]. Those hulls, in turn, can receive view-dependent texturing [Vedula et al. 2005] so that they look reasonably realistic when viewed from other angles. Careful fitting of kinematic models inside the visual hull [Carranza et al. 2003], or of 3D body scans to the outside [de Aguiar et al. 2008], or both [Vlasic et al. 2008; Ballan and Cortelazzo 2008], makes these Free Viewpoint Videos capable of representing the actor as temporally coherent in 3D. The newest results in this direction [Hasler et al. 2009], demonstrate such model-based tracking with moving cameras outdoors. With fewer priors on shape and therefore at risk of topology changes, [Starck and Hilton 2007] have produced stunning results, even for actors with loosely fitting clothing.

For Free Viewpoint Video of outdoor scenes, soccer/football games seem to be a favorite application domain [Hayashi and Saito 2006]. The broadcast-quality cameras are mostly fixed or calibrated based on known coordinates of painted field-markings, but are certainly far apart and subject to complicated outdoor effects. The largely uniform green fields help with segmentation, but as discussed in the most recent work from [Kilner et al. 2007], segmentations are usually only approximate. Their work is notable for simultaneously segmenting and modeling many players on the field who further, project to only $\sim$20 pixels in height. At that resolution, there is some opportunity for stereo-based matching of gross appearance features. They also demonstrate that naive use of billboard models for moving players produces very noticeable artifacts, even in the constrained planar world of sports fields. Our proposed work has an alternative treatment of billboards that even works in somewhat cluttered environments.

Static Object Techniques Our problem domain specifically deals with navigating and presenting the footage of *motion* captured by multi-view cameras. Nevertheless, techniques for 3D reconstruction of static scenes are also relevant. For example, [Pollefeys et al. 2004], [Lhuillier and Quan 2005], [Campbell et al. 2007], and [Goesele et al. 2007] show how well hand-held videos or photos like ours can be sufficient to reconstruct sections of a complex 3D scene. [Seitz et al. 2006] is a good survey of static-model reconstruction algorithms used mostly in idealized conditions, but complementary research continues on 3D image-based modeling techniques that leverage human effort. From [Debevec et al. 1996] through [van den Hengel et al. 2007] and now [Sinha et al. 2008] or Google Sketch-up's PhotoMatch, it is increasingly possible to semi-automatically produce static scene surface geometry both indoors and out. Our algorithm takes advantage of this, so that after-the-fact navigation of an event can be placed within the spatial context of the surrounding environment.

3 Scenery & Offline Processing

Background Scene Reconstruction Our system uses a 3D reconstruction of the static background scene to: i) provide context while rendering transitions, ii) calibrate the camera poses for each video frame, and iii) refine each camera's video matte. A variety of 3D vision methods exist for static scene reconstruction. Aside from photos and videos of a specific event, one could also use online photo collections of specific places to build dense 3D models [Goesele et al. 2007]. We follow the same structure from "motion" strategy as [Snavely et al. 2006], matching SIFT features [Lowe 2004] between photos, estimating initial camera poses and 3D points, and refining the 3D solutions via bundle adjustment. However from there, we proceed by computing a depthmap for each photo using standard multi-view planesweep stereo based on normalized cross-correlation. The final polygonal surface mesh is generated using the robust range image fusion of [Zach et al. 2007]. A static texture for the background geometry is also extracted from the photos and baked on. Since the background scene is fairly dynamic in places, much of that texture will be replaced during the interactive stage of the system, by sampling the view-dependent colors opportunistically from each camera's video.

3.1 Camera Poses

The system computes camera poses for the video frames relative to the reconstructed background scene. We refer to I_t^A as the real image seen by camera A at time t. We estimate the intrinsics, $K_t^A \in \mathbb{R}^{3\times3}$, and the extrinsics $E_t^A = [R_t^A|T_t^A]$, composed of a 3×3 rotation R_t^A and 3×1 translation T_t^A. SIFT features, found in images that had been used to reconstruct the background, are searched for potential matches to features found in each video frame. When a successful match is found, that 2D feature m in a frame of camera A corresponds to a 3D point p in the reconstructed geometry, and has the relationship $[zm, z]^T = K_t^A E_t^A p$, which holds up to the unknown depth z. The linear solution for K_t^A and E_t^A is the Direct Linear Transform (DLT) [Hartley and Zisserman 2000], and 6 matches are sufficient to solve for it.

The DLT does not guarantee that the poses of different cameras are

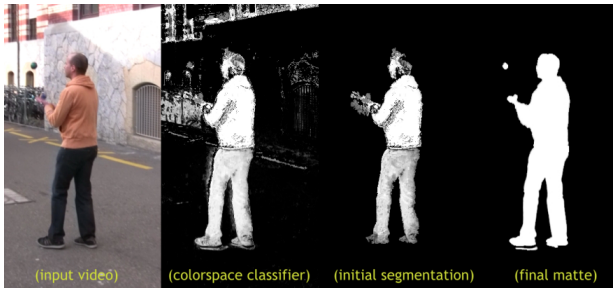

Figure 2: *Illustration of different segmentation steps for a frame of the Juggler sequence.*

recovered with the same 3D accuracy. Similar reprojection errors of sparse features, as measured in pixels, could indicate very different qualities of pose-estimation, especially when depths and resolutions vary greatly. While this is a known problem for video-based reconstruction, our VBR system can cope with this limitation. The key is to achieve a calibration that *looks* correct when the textured geometry is rendered in conjunction with the performer during the interaction stage, even if it is off by a few meters. Treating the calibration so far as an initialization, we perform a second optimization of camera poses. We use particle filtering (see [Arulampalam et al. 2002]) to minimize the sum-of-square-difference (SSD) between each I_t^A and our render-engine's versions of the reconstructed and textured scene in different poses. In this case, the texture is obtained as the median reprojected texture from a temporal window of 1000 frames of the same camera A (subsampled for efficiency).

3.2 Initial Segmentation

Segmentation in complex environments is an ongoing challenge, particularly when camera motion and moving backgrounds are expected. In our system, the user need only paint the pixels of two random images from each video camera with the binary labels $L \in \{1, 0\}$, to indicate foreground pixels that belong to the performer, *vs.* background pixels that do not. With multiple videos, each lasting potentially thousands of frames, all subsequent segmentation, both here and in Section 3.3, is computed automatically, despite the obvious complications for our VBR. Even using a primitive paint program, the user effort does not exceed 10 min. per input video. Video SnapCut [Bai et al. 2009] and Video Cutout [Wang et al. 2005] have nice interfaces allowing one to walk through and potentially correct each frame. Background Cut [Sun et al. 2006] is most effective for stationary cameras or at least static background colors. We expressly focus on changing scenes, and cannot afford to have a user check each frame.

The user-labeled training pixels define a foreground and a background color model. We simply use a k-nearest-neighbor classifier ($k = 60$) in RGB space, so the pixel-wise independent posterior probability is $\frac{k_L}{k}$, amounting to the fraction of a pixel's color neighbors that had been labeled L. To get a conservative foreground mask γ_t^A and to compute it efficiently, we store the class-conditional likelihood ratio of foreground to background in a discretized 256^3 color cube lookup table. The table usually takes 5 min. to compute, and each frame is then segmented in 2-3 sec., using 0.6 as the necessary distance ratio to label a pixel as foreground (see Figure 2). To get a conservative foreground mask during the initial segmentation step, mean-shift tracking was used to predict the area of the foreground pixels. Only pixels labeled as foreground and belonging to that area are considered foreground objects. This decreases the number of false positive foreground pixels.

3.3 Matting Through Adaptive Scene Rendering

The quality of the initial segmentation is insufficient for our rendering purposes (see Figure 2). To improve it, our matting process includes a new background color model, the same foreground color model as in Section 3.2, and graph cuts [Boykov and Kolmogorov 2004] to optimize the boundary. Each image I_t is treated as a moving foreground f_t with changing background b_t. By the compositing equation, $I_t = \alpha_t f_t + (1 - \alpha_t) b_t$, where α_t is the per-pixel alpha matte. With a binary initial segmentation γ_t in hand, we now seek to estimate f, b and a refined α for each frame. The Video Matting approach of [Chuang et al. 2002] is attractive because it produces high-quality mattes for moving cameras. However, their assumptions about users being able to spend significant time with each video (5 min. per 100 frames) and treatment of the background as a planar homography do not apply to our situations. We knowingly trade matte quality for i) less user interaction (none beyond what was done for the Initial Segmentation) ii) 3D background scenes and camera motion, and iii) potentially significant motion of people and objects positioned between the foreground and the background. Our downstream rendering process is designed specifically to cope with our lower-quality mattes.

A per-pixel color model for the background b_t of each video frame is estimated first. Dilation of the initial segmentation γ_t by 10 pixels gives a conservative background mask, removing the need for a manually specified traveling garbage matte. Knowing both the background geometry and the calibration parameters, we can render the "empty" scene seen at time t from camera A using the colors from elsewhere in A's timeline. In one sense our approach is similar to that of [Rav-Acha et al. 2008], where a model of the background is generated and textured using the input video. Here, much like Chuang et al., we determine the probability distribution of b_t by sampling from temporally proximate frames. Our algorithm collects samples of $b_t(m)$ for m's which are not labeled as foreground at time t, *i.e.,* those where $\gamma_t(m) = 0$. Further samples are collected by searching backward and forward in time with increasing Δ, projecting the images $I_{t\pm\Delta}$ with their related $\gamma_{t\pm\Delta}$, onto the scene, according to A's calibrations. Once 10 samples for the same pixel m have been collected, a Gaussian is fitted to model $b_t(m)$, though we save the medians instead of the means. This procedure has been parallelized and runs with GPU acceleration.

We first solve the compositing equation assuming α's are binary, leading to a trimap that is ready for further processing. Graph cuts is applied to maximize the conditional probability $P(\alpha_t|I_t)$, which is proportional to $P(I_t|\alpha_t)P(\alpha_t)$. Applying the logarithm and under the usual assumptions of conditional independence, $\log(P(\alpha_t))$ represents the binary potential, while $\log(P(I_t|\alpha_t))$ represents the unary potential. For each pixel m in I_t,

$$P(I_t(m)|\alpha_t) = P(f_t)^{\alpha_t(m)} P(b_t(m))^{(1-\alpha_t(m))} \quad (1)$$

where $P(f_t)$ is the foreground color model estimated in Section 3.2 and $P(b_t(m))$ is the aforementioned Gaussian distribution. Due to the inevitable presence of small calibration and background geometry errors, the projection of $I_{t\pm\Delta}$ can be imperfect by some small local transformations. To account for this, $P(b_t(m))$ is actually considered to be the maximum of all the pixels in a 5×5 neighborhood. The binary potential is formulated as the standard smoothness term, but modified to take into account both spatial and temporal gradients in the video. Once this discrete solution for α_t is found, a trimap is automatically generated by erosion and dilation (3 and 1 pixels respectively). For all grey pixels in the trimap, we apply the matting technique proposed in [Chuang et al. 2001].

It should be noted that the background subtraction/matting technique presented here only uses information from a single camera.

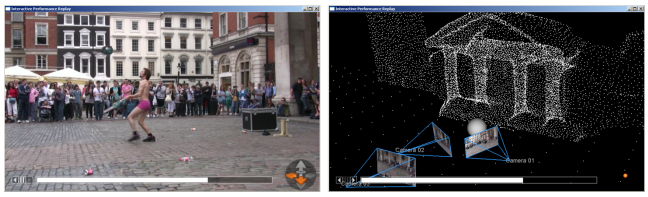

Figure 3: *Interactive Navigation Interface: Regular Mode (left) is a live preview of the content being rendered to the final output video. Orbit Mode (right) has the same functionality, but also depicts the scenery, performer, and moving cameras. Users can switch between the two modes, and always have jog/shuttle control over both the timeline of the input footage. Please see the video for a demonstration.*

While there are obvious benefits from information of other cameras, the requirements for precise geometric and photometric calibration make it challenging to improve on our present results. This is also the main reason we abstain from attempting 3D reconstruction of the dynamic elements of the scene, but prefer to use planar proxies for the foreground during transitions. When, in addition to the foreground object of interest, other elements appear in the scene, *i.e.,* the middleground, the presented segmentation procedure also extracts those. The mean-shift tracker of Section 3.2 is able to distinguish such elements from the object of interest so that during rendering, those elements are modeled as separate billboards, *i.e.,* objects of interest but with optional blur, and without focus stabilization. When instead, the 3D position of a middleground element cannot be triangulated, as happens when it appears in only one camera, our system makes it disappear before a transition starts, and reappear as the transition concludes. This situation can be observed in the Magician and the Juggler sequences, when people stand in front of somebody's cameras.

4 Online Navigation

Having precomputed a hybrid representation of the event of interest, we now present our real-time online navigation tool which allows a user to interactively explore the event from multiple viewpoints. The hybrid representation of the performance, so far, encapsulates i) static surface geometry for the scenery, ii) the surface geometry's view-independent texture, iii) time-varying camera poses, and iv) segmentations of the actor in every frame. These elements were prepared offline, in order that subsequent interactions and rendering could be real-time.

The largely GPU-driven user interface of the system lets the user smoothly navigate the video collection in both space and time. The GUI can be operated in two different modes, Regular Mode and Orbit Mode, where the same jog/shuttle and camera-transition commands are available by keyboard or mouse at all times. Those commands can be recorded and used as an edit-list for more elaborate postprocessing of an output video. The Regular Mode is essentially a rendering of the event from either a real cameras's perspective, or the virtual camera's transition when the user clicks on the navigation arrows (see Figure 3). The navigator icon on the lower right corner of the interface, indicates the possible directions that the user can go (up, down, left, right, forward and backward depending on the availability of nearby videos). Each camera's neighbors are determined relative to its image plane. Orbit Mode has a live preview window to the side, and serves primarily as a digital production control-room, where the scenery, performer, and all the moving cameras are depicted as elements of a dynamic 3D world. Orbit Mode also has a video wall option where inset views of each camera are fixed in place on the screen, but some users preferred when these individual videos played as moving screens inside the scene.

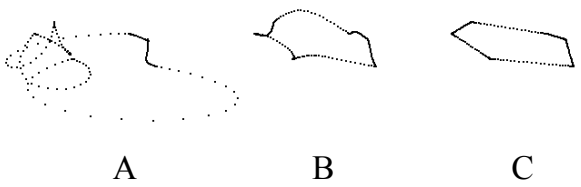

Figure 4: *Movement of the subject's baricenter through six transitions in the image for linear-SLERP (A), cylindrical-SLERP (B) and our proposed approach (C) (see Section 4.1).*

The user always watches the scene from the point of view of the *Virtual Camera* V. In our system, the virtual camera intrinsic parameters K^V are assumed to be fixed and equal to one of the cameras recording the scene. Its extrinsic parameters E_t^V are always locked to one of the cameras of the collection and unlocked only during a transition from one camera to another. From the user point of view, between user-interactions, the currently selected camera's video plays onscreen without modification. In reality, the video is playing as a texture-mapped billboard fixed relative to its camera. In this section, we introduce our billboard representation and discuss how the transition is optimized to minimize visual artifacts.

4.1 Performer-Specific Virtual Camera Path

When a camera change is requested, a virtual camera V performs the view interpolation from a starting camera A to an ending camera B, over a period of time $[t_0, t_1]$. $E_{t_0}^V = E_{t_0}^A$ at the beginning of the transition, and $E_{t_1}^V = E_{t_1}^B$ by the end.

The naive approach for computing the intermediate R's and T's is to interpolate linearly between the two projection centers, and to interpolate the camera's orientation using Spherical Linear intERPolation (SLERP). As our attention should be focused on the subject, it seems obvious to use a cylindrical interpolation centered on the intersection of the billboards. This approach has also been exploited in [Snavely et al. 2008] and achieved good results because the focus of attention is fixed in the center of the image, while the rest of the world, including our point of view, orbits around.

Our situation is different as there is no guarantee that the performer is centered in either image $I_{t_0}^A$ or image $I_{t_1}^B$. The previously mentioned techniques generate annoying artifacts here because the virtual camera's motion tries to follow parabolic paths (see Figure 4A and 4B). We still do a cylindrical interpolation of the camera projection centers, but instead of using SLERP for the rotations, we force the camera to maintain a constant and linear translation of the image of the actor (see Figure 5 and Figure 4C). Formally, given the point in space p_t, representing the barycenter of the actor, we force the image of this point in the virtual camera, at any time $t \in [t_0, t_1]$, to be the exact linear interpolation between what it was at time t_0 (in view A) and what it will be at t_1 (in view B).

Let $\Pi_t(\cdot)$ denote the projection map of the virtual camera V at time t, *i.e.,* the function mapping a generic point p in space to its projection onto the virtual camera image plane at time t. In homogeneous coordinates, $\Pi_t(\cdot)$ is subjected to the following

$$z\left[\Pi_t(p), 1\right]^T = K^V\left(R_t^V p + T_t^V\right) = K^V R_t^V\left(p - O_t^V\right) \quad (2)$$

where z is an unknown scalar and $O_t^V = -\left(R_t^V\right)^{-1} T_t^V$ represents the virtual camera origin at time t. Our goal is to generate a virtual camera path from camera A to camera B. It must satisfy the condition that for each $t \in [t_0, t_1]$, the coordinates of $\Pi_t(p_t)$ must

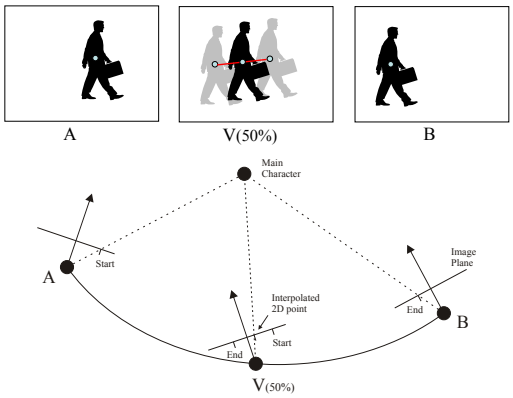

Figure 5: *A virtual camera transitions from the real camera A to the target real camera B. The rotation of the camera is defined such that the projection of the center of the main character moves along a straight line in the image. Note that the movement of the main character in the image is due to both the movement of the character in space as well as the movement of the virtual camera throughout the transition.*

be the linear interpolation of $\Pi_{t_0}(p_{t_0})$ and $\Pi_{t_1}(p_{t_1})$, *i.e.,*

$$\Pi_t(p_t) = \alpha \Pi_{t_0}(p_{t_0}) + (1-\alpha)\,\Pi_{t_1}(p_{t_1}), \qquad (3)$$

where $\alpha = (t-t_0)/(t_1-t_0)$. We assume that O_t^V is cylindrically interpolated between $O_{t_0}^V$ and $O_{t_1}^V$. Our goal is to compute a R_t^V that aligns the vector $\left(p_t - O_t^V\right)$ to $\left(K^V\right)^{-1}[\Pi_t(p_t),1]^T$. The Orthogonal Procrustes method [Schönemann 1966] can be used on the normalized versions of these vectors to obtain R_t^V, up to one degree of freedom of rotation around $\left(K^V\right)^{-1}[\Pi_t(p_t),1]^T$. That degree of freedom is fixed by linearly interpolating the angles obtained for view A and view B at the ends of the transition.

4.2 Billboard Model

Between user-interactions, the video is playing as a texture-mapped billboard fixed relative to the current real camera. As soon as a camera-change is requested from camera A to B, the foreground actor is modeled by the proxy shape of two billboards (the same is done for all the other dynamic elements of the scene, *i.e.,* the middleground). Both the billboard ζ^A and ζ^B continue to face their respective cameras. ζ^A and ζ^B are positioned at depths that make them coincide at one line in 3D space (see Figure 6). The depth of that line is computed by imposing that its projection contains the barycenters of the subject in both views. A billboard's appearance is defined by backward mapping to find the texture coordinates in the video and segmentation frames.

A billboard approximates the actor's geometry using a planar proxy, and so can introduce significant artifacts while one navigates between cameras. We propose that for lack of a better proxy, billboards can actually be quite effective, as long as we use them in tandem with a good measure of the expected visual disturbance. Ideally, while V is traveling along its path between A and B, V would cross-fade imperceptibly from rendering mostly the billboard ζ^A to showing mostly ζ^B. We have observed that a well-placed billboard is a convincing enough proxy shape for viewing-angle changes around 10°, but the illusion can quickly be lost when the second billboard comes into view, especially gradually, as is the case with a linear cross-fade. The enhanced Cross Dissolve of [Grundland et al. 2006] could help, but we have found that if timed correctly, a cut from one billboard to the next can be almost unnoticeable. [Schödl et al. 2000] made a similar observation.

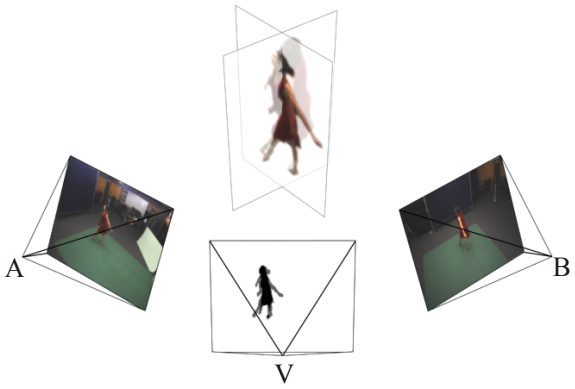

Figure 6: *As the virtual camera transitions from view A to view B, the foreground object is represented by two video sprites on planar billboards, one for each view. The video footage from each camera is rendered onto the respective billboard with the segmentation mask applied.*

It is preferable for the user to confuse a sharper appearance change-over with the performer's natural ongoing motions. The best time for appearance change-over is when the *action* is at its most fronto-parallel to the two cameras. Choosing a bad time will reveal the actor's current 3D shape as non-planar. Section 4.3 explains the simple strategy that finds the best change-over time, but first we introduce the error measure to be optimized.

The Inter-Billboard Distance at time t for camera V is computed using the following procedure. Each billboard, ζ^A and ζ^B, is first rendered separately from the viewpoint of the virtual camera V at time t using the masks α_t^A and α_t^B as texture. Those two images are then thresholded, producing two silhouette images, S_1 and S_2. Overlaying S_2 on S_1, as pictured in the V camera of Figure 6, allows one to evaluate how much change a user can perceive if the two billboards are suddenly swapped during the transition. The change is less perceptible the more these two images agree. To quantify this agreement, we use the distance measure

$$D(S_1,S_2) = \frac{1}{\#S_1}\sum_{m\in S_1} d(m,S_2) + \frac{1}{\#S_2}\sum_{m\in S_2} d(m,S_1), \quad (4)$$

where m represents a pixel inside the silhouette, and $d(m,S)$ is the l^2-distance between this point and a silhouette S. $\#S_1$ and $\#S_2$ represent the numbers of points in silhouette S_1 and S_2, respectively. This error can be quickly computed in a fragment shader using the distance transform [Rong and Tan 2006]. We also tried a correlation-based distance, but found it less effective at matching the perceptual differences observed by the user. In fact, changes of appearance within the silhouette that occur during a change-over are often perceptually confused with subject motion.

4.3 Transition Optimization

The Inter-Billboard Distance (4) largely dictates the right moment to switch billboards. We have found that also including the start time as a parameter can lead to a much better optimum. Thus we optimize over two variables: ρ which is the fraction of the transition interval at which the billboard transition occurs, and Δ which is the transition delay, the time between the user's request and its actual start. This search is similar in spirit to the approach [Wang and Bodenheimer 2008] proposed for combining motion capture data. The start time is delayed by no more than 3 sec., and the transition time was set to 1.5 sec. Since the search domain is limited and known, a fast grid search optimizes both parameters in a separate thread to preserve real-time playback.

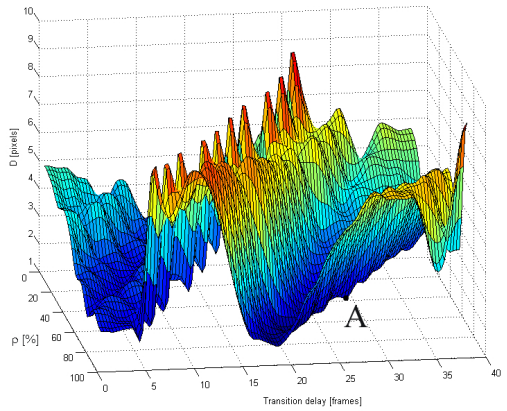

Figure 7: *During a transition, the moment at which to switch from rendering one billboard to rendering the next is computed by a grid search optimization. The parameters are ρ the fraction of the transition interval at which to switch, and Δ the (small) amount of time to delay the transition start time. In this case A is the optimum.*

Figure 7 illustrates the Inter-Billboard distance evaluated at a certain time in the Juggler sequence. Notice that the graph exhibits a clear diagonal structure. This is because the points in the diagonally aligned valleys are points corresponding to the same billboard transition frame. Thus in this case, there is a clear moment at which the billboard transition should occur.

4.4 Rendering

During normal playback of a video in Regular Mode, the virtual camera position is locked to the real camera's extrinsics. Depending on camera A's intrinsic parameters K_t^A, the original video is played at a different size in relation to the virtual camera intrinsics K^V. Black borders are added to the video if its size is smaller than the virtual camera. This happens, for instance, when one camera has landscape orientation while the other is in portrait mode, or if the zooms are different. While it is possible to adapt the intrinsic camera parameters of the virtual camera to those of the real one, that can create perceptually undesirable effects (*i.e.,* the *Vertigo* effect).

Once the user requests a transition, the exact timing is optimized as described in Section 4.3. The transition is performed to minimize disturbing visual artifacts. During the first 20% of the transition, the virtual camera remains locked to the original viewpoint, but the scene rendering fades from the original video to the synthetically rendered scene (at which point the black borders disappear). Then the virtual camera starts moving along the path defined in Section 4.1 while the video is still playing. Like the start of the transition, the virtual camera is locked to the target camera position for the last 20% of the transition, when video of the target camera fades in. During the entire transition, the video is rendered using the color space of the original camera. This is done by using precomputed 3×3 color transformations, approximately mapping the appearance between videos, and also from the view-independent texture to the videos. Only during the last 20% is the appearance gradually transformed to the target video.

Next, the middle of the synthetically rendered video transition is created. Although a very large amount of footage available for rendering, a real-time rendering application must take bandwidth and other system hardware limitations into account. Using all the available videos, masks, and background videos simultaneously would require far too many resources to render the scene interactively.

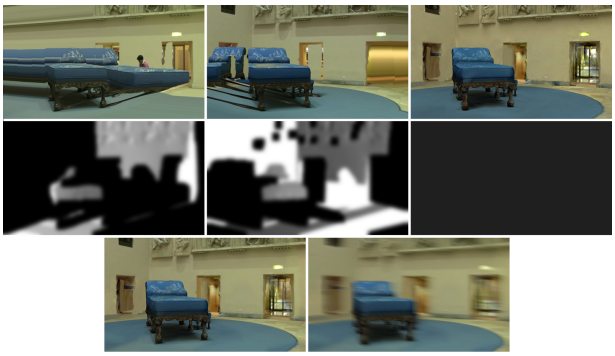

Figure 8: *Background rendered from left, right, and view-independent texture (top), corresponding suitability maps (middle), final rendered background and generated motion blurred background (bottom).*

To render a transition from camera A to camera B, we chose to load and use only data extracted from videos A and B, and the static information of the scene. These two cameras are normally also the closest to the virtual camera path, and the benefits of using more videos are often limited. This tradeoff is similar to the one made for IBR of static scenes by [Debevec et al. 1998], where at most three views were used to texture each scene element.

We adapt the Unstructured Lumigraph Rendering framework [Buehler et al. 2001] to cope with the fact that some parts of the background scene are occluded by foreground and that we can only afford to use two videos. At each time t, the images I_t^A and I_t^B are used to color the geometry of the background scene, as in [Buehler et al. 2001]. We generate α-masks from the α_t^A and α_t^B associated with the foreground, to mask the foreground pixel elements of I_t^A and I_t^B, respectively. Three images of the scene from the point of view V are generated: the first one uses only the color information from I_t^A, the second uses colors from I_t^B, while the last one uses the view-independent texture extracted in the pre-processing stage. The view-independent texture is necessary because on the path between A and B, the virtual camera can see parts of the scene that are hidden in both A and B. For each generated image, a per-pixel suitability mask is generated in parallel, taking into account the α-masks (*i.e.,* that a pixel is background or not), occlusions, and viewing angles. Occlusions are handled by rendering the depth maps of both I_t^A and I_t^B. We use the angle differences with respect to the surface normals to weight each pixel from the two sources. This is important in the presence of miscalibrations and geometry errors.The suitability mask of the image generated using the static texture is given a constant low value so that its colors are used only where neither of the other images can provide useful information. After the suitability has been computed, a dilation/erosion and smoothing filter is applied to ensure a spatially smooth transition between the texture during the blending, and to account for discontinuities and blobs that can appear due to occlusion handling and matting errors. The entire procedure is implemented on the GPU using a 3-pass rendering. As a final step, a motion blur filter is applied to all the pixels belonging to the background scene, which makes the foreground object stand out. This is a user-controllable option in the software, and we found that, when enabled, the user's attention is focused on the performer, *i.e.,* the center of the action, while the motion blur gives peripheral cues about the direction and the speed of the transition. The whole background rendering approach is illustrated in Figure 8.

The foreground elements of the scene are then rendered using a

similar technique with the images I_t^A and I_t^B, and the appropriate alpha masks α_t^A and α_t^B. Only the two billboards ζ^A and ζ^B are rendered for each foreground element. The transition from ζ^A to ζ^B is decided as described in Section 4.3.

5 Results

There are many casually filmed events, but multi-view footage that is public-domain is so far, readily available only when Citizen-journalists are provided with a specified portal for submissions *e.g.,* after a U2 concert. We obtained the Climber and Dancer datasets from [Hasler et al. 2009] and INRIA Grenoble Rhone-Alpes respectively, and the Juggler, Magician, and Rothman data by attending real events, handing out cameras to members of the public with instructions to play with the settings, and where needed, obtaining signatures allowing for use and dissemination of the footage. These events were chosen because together, they explore a variety of challenges in terms of inter-camera distance, large out-of-plane motion, fast performances of skill, complicated outdoor and indoor lighting conditions, and intrusive objects in the field of view. We performed the manual part of the process ourselves, labeling the performer in two frames per video, and locating ~ 40 12MP photos of each new environment. The Climber videos are 720×544, the Dancer videos are 780×582, and the new footage measures 960×544 pixels, with people filming in landscape or portrait mode (or switching), with different settings for zoom, automatic gain, and white balance. Some people adjusted these manually at times.

Naturally, the results of this interactive VBR system are best evaluated in video, so please see http://cvg.ethz.ch/research/unstructured-vbr/. Among the videos, several demonstrate specific stages of the algorithm such as rendering-for-matting, and several show events produced by volunteer test-subjects.

Similar colors on the performer and the background are inevitable, which our Initial Segmentation confirms repeatedly. Even drastically increasing the amount of training data had no effect. The γ masks are frequently exaggerated in size, but that being only an intermediate stage, simply meant that the Adaptive Scene Renderer had to seek further out in the timeline to obtain enough samples. With our new form of background subtraction, even significant imperfections in the reconstructed scene geometry did not hinder us from pulling a useful matte, probably because those imperfections coincided with textureless areas. The bigger segmentation problems occur when the subject exhibits significant motion blur, because mixed pixels can match the rendered background quite well.

Clutter in the scene is caused by both objects that change and people who move around the performer. Theoretically, enough moving cameras could allow 3D visual hull reconstruction for the Juggler, if our pipeline were followed through Section 3.3. However, while scenes like Magician and Rothman have enough cameras in positions to triangulate billboards (of the performer and the clutter), their coverage is sparse and their calibrations and segmentations are off by too much to yield acceptable 3D shapes. We also experimented with computing heightfields to augment our billboards, but without structured lights like those of [Waschbüsch et al. 2007], the results were disappointing. These findings seem consistent with [Kilner et al. 2006]. The modicum of clutter in the scenes we tested was handled with relatively few artifacts because elements that were rejected from the background model either ended up as middleground billboards due to their 3D separation from the performer, or when incorrectly merged with the performer in one view, were deemed too costly by the Transition Optimization.

The prototype system is real-time, running at 25 fps on an Intel i7 2.93Ghz Quad-core with 8GB of memory, an nVidia GTX285 GPU, and a RAID0 with four SSD hard drives. Even events filmed with at least six cameras can be explored without impacting performance, because videos are streamed locally, and can be subsampled if HD footage were available. The information necessary for the next frames are preloaded by a separate thread to allow undisturbed real-time rendering. An optional post-processing stage can be triggered after the user has recorded their intended interactions. Not shown here, but this renders a higher-quality composite, with audio, from the original source videos, which are heavily compressed by at least our Canon HG10's, but are non-ideal for streaming. In Figure 10, three of our many example transitions are shown between different cameras.

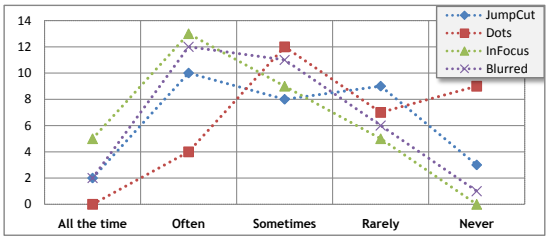

Figure 9: *# of users with each preference: 32 users reported how often they would use each transition type. Dots, InFocus, and Blurred refer to the three styles used to render the background geometry while transitioning.*

32 volunteers were asked to use the prototype system while navigating 3 prepared video collections captured at different performances. Users varied in experience, having from none to extensive practice with video editing. Each user received 4 minutes of instruction, a printed list of the possible navigation and rendering modes, and eventually filled in a questionnaire. The overall response was quite positive, and Figure 9 shows the responses when users were asked to evaluate how often they would switch between videos using the available transitions. Of the available navigation modes, 4 preferred Regular Mode, 3 preferred Orbit Mode, and 25 liked both.

6 Discussion

The benefit of using our multi-view VBR algorithm rests in the added editorial value of this new system. This new framework is unique in giving interactive control over what would normally be a collection of one-at-a-time hand-held videos of a performance. A few superb algorithms can already stabilize casually captured footage [Liu et al. 2009], or re-synthesize moving actors filmed in studio conditions or with fixed and narrow camera baselines. However, our system substantially increases the domain of usable footage, running, to our knowledge, on the most difficult sequences considered thus far for input to VBR.

The contribution of our technique is that those difficult videos are combined into an interactive representation that can be navigated along visually realistic paths. The user-navigation is simple, but sufficient to navigate among the available cameras while letting the user keep track of the overall 3D environment. The spatial awareness offered by the interface is carried over into the system's output video, which renders camera viewpoint changes by providing smooth visual transitions of the background scene, even across big angular and spatial separations.

The main insight from this work is that it is possible to design some VBR applications with mechanisms for coping with flawed input, such as our optimization of inter-billboard distances. Our rendering-based video matting and pose-estimation algorithms attempt to improve on a hard situation, and while user-intervention

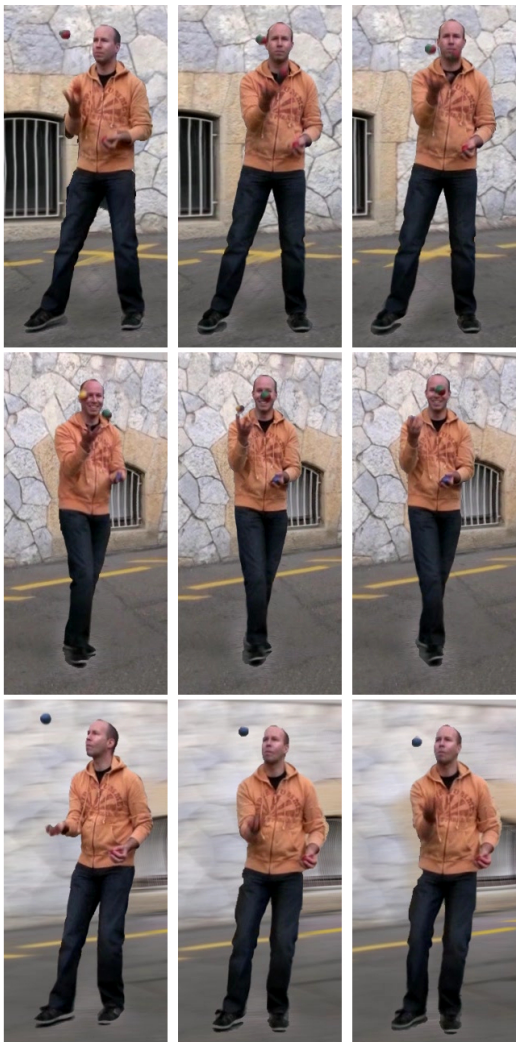

Figure 10: *Each row shows a different transition. The three consecutive frames span the best changeover (i.e., switch between billboards) found within a given timeframe. The optimization is successful if this changeover is hard to perceive. The background can be motion blurred or not.*

is the normal way of "repairing" problems, this quantity of data requires explicit bad-situation avoidance.

Limitations Some of the limitations of our system are obvious, such as subtle segmentation problems near object boundaries, that appear as shimmering during the transitions, or motion-blurred objects which we cannot segment correctly at all. Further, it would be nice to segment the videos with even less user-input. In the future, we would hope to find ways to cope with multiple performers that come close to each other in 3D, possibly by learning the statistics of each actor's appearance. Since our segmentation currently relies on the scene reconstruction stage, it could be interesting to leverage both the spatial 3D and temporal correlation to segment each frame in a multi-view sequence *jointly*. Certainly, future improvements in video matting could also be incorporated into our system, as our main goal has been to provide a proof of concept, and certain parts can be replaced. Each scene's geometry takes $\sim$1hr to reconstruct and is automatic except that part way through, the [Zach et al. 2007] pipeline requires the user to designate a bounding box for the volume reconstruction, and afterwards fit a plane for the ground. After this user effort, the automatic processing takes multiple hours, as detailed in the supplemental materials. Semi-/Automatic geometric reconstruction of even static scenes continues to be an important research challenge.

In summary, with less than 30min. training and a small amount of user-input, our approach converts a collection of hand-held videos into a digital performance that can easily be navigated and re-rendered in a way that was previously impossible.

Acknowledgements We thank Ralph Wiedemeier, Davide Scaramuzza, and Mark Rothman whose performances constitute the Juggler, Magician, and Rothman data, Nils Hasler and Jürgen Gall for the Climber videos, and Christopher Zach, David Gallup, Oisin Mac Aodha, Mike Terry, and the anonymous reviewers for help and valuable suggestions. The research leading to these results has received funding from the ERC under the EC's Seventh Framework Programme (FP7/2007-2013) / ERC grant #210806, and from the Packard Foundation.

References

ARULAMPALAM, M. S., MASKELL, S., AND GORDON, N. 2002. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. Signal Processing 50*, 174–188.

BAI, X., WANG, J., SIMONS, D., AND SAPIRO, G. 2009. Video snapcut: robust video object cutout using localized classifiers. *ACM Trans. Graph. 28*, 3.

BALLAN, L., AND CORTELAZZO, G. M. 2008. Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. In *3DPVT*.

BOYKOV, Y., AND KOLMOGOROV, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 9, 1124–1137.

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S. J., AND COHEN, M. F. 2001. Unstructured lumigraph rendering. In *SIGGRAPH*, 425–432.

CAMPBELL, N. D., VOGIATZIS, G., HERNÁNDEZ, C., AND CIPOLLA, R. 2007. Automatic 3d object segmentation in multiple views using volumetric graph-cuts. In *18th British Machine Vision Conference*, vol. 1, 530–539.

CARRANZA, J., THEOBALT, C., MAGNOR, M. A., AND PETER SEIDEL, H. 2003. Free-viewpoint video of human actors. In *ACM Transactions on Graphics*, 569–577.

CHEN, S. E., AND WILLIAMS, L. 1993. View interpolation for image synthesis. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, 279–288.

CHUANG, Y.-Y., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2001. A bayesian approach to digital matting. In *Proceedings of IEEE CVPR 2001*, vol. 2, 264–271.

CHUANG, Y.-Y., AGARWALA, A., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2002. Video matting of complex scenes. *ACM Transactions on Graphics 21*, 3 (July), 243–248.

DE AGUIAR, E., STOLL, C., THEOBALT, C., AHMED, N., SEIDEL, H. P., AND THRUN, S. 2008. Performance capture from sparse multi-view video. *ACM Trans. Graph. 27*, 3, 1–10.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, 11–20.

DEBEVEC, P., BORSHUKOV, G., AND YU, Y. 1998. Efficient view-dependent image-based rendering with projective texture-mapping. In *9th Eurographics Workshop on Rendering*.

DRAGICEVIC, P., RAMOS, G., BIBLIOWITCZ, J., NOWROUZEZAHRAI, D., BALAKRISHNAN, R., AND SINGH, K. 2008. Video browsing by direct manipulation. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, 237–246.

EISEMANN, M., DECKER, B. D., MAGNOR, M., BEKAERT, P., DE AGUIAR, E., AHMED, N., THEOBALT, C., AND SELLENT, A. 2008. Floating Textures. *Computer Graphics Forum (Proc. Eurographics EG'08) 27*, 2 (4), 409–418.

FRANCO, J.-S., AND BOYER, E. 2005. Fusion of multi-view silhouette cues using a space occupancy grid. In *ICCV*, 1747–1753.

GOESELE, M., SNAVELY, N., CURLESS, B., HOPPE, H., AND SEITZ, S. M. 2007. Multi-view stereo for community photo collections. In *ICCV*, 1–8.

GOLDMAN, D. B., GONTERMAN, C., CURLESS, B., SALESIN, D., AND SEITZ, S. M. 2008. Video object annotation, navigation, and composition. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, 3–12.

GOLDMAN, D. B. 2007. *A framework for video annotation, visualization, and interaction*. PhD thesis.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The lumigraph. In *SIGGRAPH*, 43–54.

GRUNDLAND, M., VOHRA, R., WILLIAMS, G. P., AND DODGSON, N. A. 2006. Cross dissolve without cross fade: Preserving contrast, color and salience in image compositing. In *Proceedings of EUROGRAPHICS, Computer Graphics Forum*, 577–586.

GUILLEMAUT, J.-Y., HILTON, A., STARCK, J., KILNER, J., AND GRAU, O. 2007. A bayesian framework for simultaneous matting and 3d reconstruction. In *3DIM '07: Proceedings of the Sixth International Conference on 3-D Digital Imaging and Modeling*, 167–176.

GUILLEMAUT, J.-Y., KILNER, J., AND HILTON, A. 2009. Robust graph-cut scene segmentation and reconstruction for free-viewpoint video of complex dynamic scenes. In *Proc. International Conference on Computer Vision (ICCV 2009)*.

HARTLEY, R. I., AND ZISSERMAN, A. 2000. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049.

HASLER, N., ROSENHAHN, B., THORMÄHLEN, T., WAND, M., GALL, J., AND SEIDEL, H.-P. 2009. Markerless motion capture with unsynchronized moving cameras. In *CVPR*, 224–231.

HAYASHI, K., AND SAITO, H. 2006. Synthesizing free-viewpoint images from multiple view videos in soccer stadium. In *CGIV '06: Proceedings of the International Conference on Computer Graphics, Imaging and Visualisation*, 220–225.

HAYS, J., AND EFROS, A. A. 2007. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007) 26*, 3.

HEIGL, B., KOCH, R., POLLEFEYS, M., DENZLER, J., AND VAN GOOL, L. 1999. Plenoptic modeling and rendering from image sequences taken by hand-held camera. In *Patter Recognition 1999, 21. DAGM-Symposium*, 94–101.

KANADE, T., 2001. Carnegie mellon goes to the superbowl. http://www.ri.cmu.edu/events/sb35/tksuperbowl.html.

KARRER, T., WEISS, M., LEE, E., AND BORCHERS, J. 2008. Dragon: a direct manipulation interface for frame-accurate in-scene video navigation. In *CHI '08*, 247–250.

KILNER, J., STARCK, J., AND HILTON, A. 2006. A comparative study of free-viewpoint video techniques for sports events. *European Conference on Visual Media Production (CVMP)*.

KILNER, J., STARCK, J., HILTON, A., AND GRAU, O. 2007. Dual-mode deformable models for free-viewpoint video of sports events. In *3DIM07*, 177–184.

KOPF, J., NEUBERT, B., CHEN, B., COHEN, M., COHEN-OR, D., DEUSSEN, O., UYTTENDAELE, M., AND LISCHINSKI, D. 2008. Deep photo: model-based photograph enhancement and viewing. *ACM Trans. Graph. 27*, 5, 116.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH*, 31–42.

LHUILLIER, M., AND QUAN, L. 2005. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 3, 418–433.

LIU, F., GLEICHER, M., JIN, H., AND AGARWALA, A. 2009. Content-preserving warps for 3d video stabilization. In *ACM SIGGRAPH 2009*, 1–9.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2, 91–110.

MATUSIK, W., BUEHLER, C., RASKAR, R., GORTLER, S. J., AND MCMILLAN, L. 2000. Image-based visual hulls. In *Proceedings of ACM SIGGRAPH*, 369–374.

POLLEFEYS, M., VAN GOOL, L., VERGAUWEN, M., VERBIEST, F., CORNELIS, K., TOPS, J., AND KOCH, R. 2004. Visual modeling with a hand-held camera. *IJCV 59*, 3, 207–232.

RAV-ACHA, A., KOHLI, P., ROTHER, C., AND FITZGIBBON, A. 2008. Unwrap mosaics: A new representation for video editing. *ACM Transactions on Graphics (SIGGRAPH 2008)* (August).

RONG, G., AND TAN, T.-S. 2006. Jump flooding in gpu with applications to voronoi diagram and distance transform. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)*, ACM, 109–116.

SCHINDLER, G., AND DELLAERT, F. 2010. Probabilistic temporal inference on reconstructed 3D scenes. In *CVPR*, 1–8.

SCHÖDL, A., SZELISKI, R., SALESIN, D. H., AND ESSA, I. 2000. Video textures. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 489–498.

SCHÖNEMANN, P. 1966. A generalized solution of the orthogonal procrustes problem. *Psychometrika 31*, 1 (March), 1–10.

SEITZ, S. M., AND DYER, C. R. 1996. View morphing. In *Proceedings of ACM SIGGRAPH*, 21–30.

SEITZ, S. M., CURLESS, B., DIEBEL, J., SCHARSTEIN, D., AND SZELISKI, R. 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, 519–528.

SINHA, S. N., AND POLLEFEYS, M. 2004. Synchronization and calibration of camera networks from silhouettes. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 1*, 116–119.

SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3d architectural modeling from unordered photo collections. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2008) 27*, 5, 159.

SIVIC, J., AND ZISSERMAN, A. 2003. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, vol. 2, 1470–1477.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, 835–846.

SNAVELY, N., GARG, R., SEITZ, S. M., AND SZELISKI, R. 2008. Finding paths through the world's photos. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008) 27*, 3, 11–21.

STARCK, J., AND HILTON, A. 2007. Surface capture for performance based animation. *IEEE Computer Graphics and Applications 27(3)*, 21–31.

STICH, T., LINZ, C., ALBUQUERQUE, G., AND MAGNOR, M. 2008. View and time interpolation in image space. *Computer Graphics Forum (Proc. Pacific Graphics) 27*, 7.

SUN, J., ZHANG, W., TANG, X., AND SHUM, H.-Y. 2006. Background cut. In *ECCV (2)*, 628–641.

TUYTELAARS, T., AND VAN GOOL, L. 2004. Synchronizing video sequences. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on 1*, 762–768.

VAN DEN HENGEL, A., DICK, A., THORMÄHLEN, T., WARD, B., AND TORR, P. H. S. 2007. Videotrace: Rapid interactive scene modelling from video. *ACM Transactions on Graphics 26*, 3 (July), 86:1–86:5.

VEDULA, S., BAKER, S., AND KANADE, T. 2005. Image-based spatio-temporal modeling and view interpolation of dynamic events. *ACM Transactions on Graphics 24*, 2 (Apr.), 240–261.

VLASIC, D., BARAN, I., MATUSIK, W., AND POPOVIĆ, J. 2008. Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics 27*, 3, 97:1–97:9.

WANG, J., AND BODENHEIMER, B. 2008. Synthesis and evaluation of linear motion transitions. *ACM Trans. Graph. 27*, 1, 1–15.

WANG, J., BHAT, P., COLBURN, R. A., AGRAWALA, M., AND COHEN, M. F. 2005. Interactive video cutout. *ACM Trans. Graph. 24*, 3, 585–594.

WASCHBÜSCH, M., WÜRMLIN, S., AND GROSS, M. H. 2007. 3d video billboard clouds. *Computer Graphics Forum (Proc. Eurographics EG'07) 26*, 3, 561–569.

WÜRMLIN, S., AND NIEDERBERGER, C., 2010. Realistic virtual replays for sports broadcasts. http://www.liberovision.com/.

ZACH, C., POCK, T., AND BISCHOF, H. 2007. A globally optimal algorithm for robust tv-l1 range image integration. In *IEEE International Conference on Computer Vision (ICCV)*.

ZITNICK, C. L., KANG, S. B., UYTTENDAELE, M., WINDER, S., AND SZELISKI, R. 2004. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics 23*, 3 (Aug.), 600–608.