Space-Time Visual Effects as a Post-Production Process

Christian Linz
Computer Graphics Lab
TU Braunschweig, Germany
linz@cg.cs.tu-bs.de

Christian Lipski
Computer Graphics Lab
TU Braunschweig, Germany
lipski@cg.cs.tu-bs.de

Lorenz Rogge
Computer Graphics Lab
TU Braunschweig, Germany
rogge@cg.cs.tu-bs.de

Christian Theobalt
Max Planck Institute Informatik
Saarbrücken, Germany
theobalt@mpii.de

Marcus Magnor
Computer Graphics Lab
TU Braunschweig, Germany
magnor@cg.cs.tu-bs.de

ABSTRACT

Space-time visual effects play an increasingly prominent role in recent motion picture productions as well as TV commercials. Currently, these effects must be meticulously planned before extensive, specialized camera equipment can be precisely positioned and aligned at the set; once recorded, the effect cannot be altered or edited anymore. In this paper, we present an alternative approach to space-time visual effects creation that allows flexible generation and interactive editing of a multitude of different effects during the post-production stage. The approach requires neither expensive, special recording equipment nor elaborate on-set alignment or calibration procedures. Rather, a handful of off-the-shelf camcorders, positioned around a real-world scene suffice, to record the input data. We synthesize various space-time visual effects from unsynchronized, sparse multi-view video footage by making use of recent advances in image interpolation. Based on a representation in a distinct navigation space, our space-time visual effects (STF/X) editor allows us to interactively create and edit on-the-fly various effects such as slow motion, stop motion, freeze-rotate, motion blur, multi-exposure, flash trail and motion distortion. As the input to our approach consists solely of video frames, various image-based artistic stylizations, such as speed lines and particle effects are also integrated into the editor. Finally, different effects can be combined, enabling the creation of new visual effects that are impossible to record with the conventional on-set approach.

Categories and Subject Descriptors

I.3.m [**Miscellaneous**]: Image- and Video-based Rendering; I.3.3 [**Picture/Image Generation**]: Viewing algorithms

1. INTRODUCTION

Recent movie releases and TV commercials give ample evidence that visual effects have become an integral part of motion picture production. Visual effects can be broadly categorized by way of production: while many effects are based on traditional 3D computer graphics technology which are created off-line, space-time visual effects, referred to in the following as STF/X, are image-based and currently need to be recorded directly on-set. Since the release of the movie "The Matrix", space-time visual effects have been used in a number of motion pictures and TV commercials. To create these effects, time-slice photography and other special recording setups are being used [5] to capture a real-world, dynamic scene in some unconventional way, e.g., to create *freeze-rotate*, *slow motion*, *motion blur*, *motion distortion* or *multi-exposure* effects. Typically, each frame of the later effect sequence is recorded by a separate camera. On set, innumerous cameras must be exactly positioned and aligned, and shutter timings of all cameras must be precisely triggered. Once captured, the only way to alter a recorded effect is to re-take the entire shot, which is why a lot of time and effort has to go into planning and capturing each space-time visual effect. In summary, the contemporary production process of STF/X requires a lot of time, money and expensive hardware.

Goal of our work is to overcome current limitations in STF/X production. We present an alternative approach that allows creating space-time visual effects as a post-production process. Our approach requires only a good handful of unsynchronized, uncalibrated camcorder recordings of essentially any real-world, dynamic scene. In separating STF/X creation from image acquisition, arbitrary visual effects can be interactively designed, edited and combined from the same recorded footage. Our image-based approach makes use of recent advances in image interpolation to estimate the plenoptic function of a dynamic scene from only a sparse set of image samples [17]. We build upon the fundamental work by Chen and Williams [4] and the graphical representation of visual effects introduced by Wolf [20] to propose a space-time navigation space that is amenable to intuitive and interactive STF/X creation and editing. Various space-time visual effects like *freeze-rotate shots*, *time/space ramps*, *slow motion* and *match cut* [5], are created by simply specifying the camera path through this navigation space. By incorporating frame accumulation rendering, our STF/X editor also features camera shutter effects like *long exposure*, *multi-exposure* and *flash effects*. As our approach relies on image data only, artistic stylization techniques such as *speed lines* and *particle effects* can be applied as well. Finally, different space-time effects can be combined in arbitrary fashion, and the resulting STF/X sequence can be viewed instantly at real-time frame rates for interactive refinement and editing.

After highlighting related work in the following section, we describe our space-time navigation space in Sect. 3 and demonstrate how to interpolate the dynamic scene's plenoptic function from the recorded multi-video footage. We go on to explain the multi-video acquisition procedure in Sect. 4 before we describe how different space-time visual effects can be realized in Sect. 5. Results for a selection of effects and a number of different real-world scenes

MM'10, October 25–29, 2010, Firenze, Italy.

are presented in the accompanying video. We discuss remaining challenges and limitations of our approach in Sect. 7 before we conclude in Sect. 8.

2. RELATED WORK

Visual effects design is an integral part of today's motion picture production. For the sake of brevity, we limit our discussion to space-time effects that are currently created using specialized multi-camera setups. Wolf [20] gives an overview of various space-time visual effects for both still and video cameras. He introduces an insightful visualization scheme for space-time effects by specifying plenoptic function samples in two-dimensional space-time diagrams. Digital Air Inc. [5] uses specialized camera hardware to record these plenoptic samples directly from real-world scenes. On set, camera positions and directions, shutter settings and exposure times must all correspond precisely to the pre-planned plenoptic samples. To alter the STF/X sequence even slightly requires a full re-take of the entire shot.

Free-viewpoint navigation systems offers a way out of this dilemma: instead of recording effect-specific plenoptic function samples, they allow to synthesize the needed samples of the plenoptic function from arbitrary, sparse sample recordings. Such systems have received considerable attention in computer graphics as well as in computer vision. Again, we limit our discussion to systems most closely related to our approach. Most similar to ours and considered as state-of-the art is the Video View Interpolation system by Zitnick et al. [23]. Based on layered, dense depth maps, this system operates on synchronized and calibrated multi-video footage which is acquired using a custom-built multi-camera system. Their approach achieves high-quality image interpolation, however, is limited to view interpolation alone. Zitnick et al. already showed the applicability of their system for visual effects generation. However, being limited to view interpolation alone, only the subset of STF/X effects not requiring arbitrary temporal samples (space blur, freeze-rotate) could be generated. Space-Time Light Field Rendering overcomes the dependence on synchronized recordings [19]. This system operates on unsynchronized video streams and is thus capable of interpolating in space and time. The input images are warped twice, first to a common virtual time before Unstructured Lumigraph Rendering [3] is applied in a second step to obtain the final result. Unfortunately, the approach requires considerable computation time and takes several minutes per output frame. Recently, Zheng et al. [22] described a system for effect choreography within the confines of a static light field. Their system allows to create 3D pan & scan effects, but mainly targets still images and a small range of viewpoints instead of video footage. All of those approaches require some sort of geometry, either in the form of dense per-frame depth maps or a geometry proxy. Their applicability for STF/X production is hence limited by acquisition and/or 3D scene reconstruction constraints.

Image interpolation allows to generate in-between images without the need for geometry reconstruction. A generally applicable, feature-based method for interpolating between two different images is presented by Beier and Neely [2]. Chen and Williams show how general image interpolation can be used for view interpolation [4]. For improved rendering performance, McMillan and Bishop propose a planar-to-planar, forward mapped image warping algorithm [12]. Mark et al. adapt the method to achieve high frame rates for post-rendering [11], while Zhang et al. apply feature-based morphing to light fields [21]. Lee et al. extended the feature-based method presented by Beier and Neely [2] to more than two images [9]. Based on visually plausible, dense image correspondence fields, warping/morphing-based image interpolation is considerably more flexible than epipolar-constrained view interpolation; calibration imprecision, unsynchronized multi-video footage, or geometry inaccuracies do not hamper correspondence-based interpolation. In addition, plausible image correspondence fields can often still be established for scenes for which depth or 3D geometry is hard to come by [18]. Recently, Mahajan et al. presented a path-based interpolation for image pairs that operates in the gradient domain and prevents ghosting/blurring and many occlusion artifacts visible in morphing-based methods [10].

3. INTERPOLATION OF THE PLENOPTIC FUNCTION

In this section, we discuss how to employ space-time view interpolation to estimate samples of the plenoptic function of a dynamic scene from only sparsely distributed video frame recordings. By treating the spatial and temporal dimensions jointly, we can continuously interpolate the plenoptic function between spatially or temporally adjacent video frames.

3.1 Navigation space embedding

Our goal is to explore a scene captured by multiple video cameras in an intuitive way and to render a (virtual) view I_v of the scene that is parameterized by a viewing direction and time instant. To this end, we choose to define a three-dimensional navigation space $\mathcal{N}$ that represents spatial camera coordinates as well as the temporal dimension. In their seminal paper, Chen and Williams [4] propose to interpolate the extrinsic camera parameters $\mathbf{R}$ and $\mathbf{p}$ directly in six-dimensional hyperspace. While this is perfectly feasible in theory, it has two major drawbacks in practice: it neither allows for intuitive exploration of the scene by a user, nor is it practical to handle the amount of emerging data needed for interpolation in this high-dimensional space. The crucial design decision in our STF/X editor is hence to map the extrinsic camera parameters to a lower dimensional space that allows intuitive navigation. The temporal dimension t already defines one axis of the navigation space $\mathcal{N}$, leaving two dimensions φ and θ for parameterizing the camera orientation and position.

Since our camera setups are not restricted to a specific configuration, the embedding has to be sufficiently flexible as well. To define our navigation space $\mathcal{N}$, we assume that we know the extrinsic camera parameters $\mathbf{R}$ and $\mathbf{p}$ for every camera, as well as scene points in 3D world coordinates. We will point out how to obtain them from the recorded multi-view footage in Sect. 4. For a specific virtual image I_v, we want to interpolate the image at a given point in navigation space

$$I_v = I_v(\varphi, \theta, t).$$

To serve as sampling points, the camera configuration of our recorded multi-video input in Euclidean world space is embedded into navigation space $\mathcal{N}$

$$\Psi : (\mathbf{R}, \mathbf{p}, t) \mapsto (\varphi, \theta, t).$$

In our approach, Ψ is a defined by fitting a bivariate polynomial $g(\varphi, \theta)$ of degree 3 to the camera positions in a least-squares sense. The cameras' viewing directions obtained from $\mathbf{R}$ provide an additional constraint on the normals of the fitted polynomial surface. The polynomial surface is then oriented such that the projection of the normal of a user defined scene ground plane onto $g(\varphi, \theta)$ is aligned with the θ-direction. The anchor point of the polynomial is defined by labeling one of the cameras as the Master camera $\mathbf{c}_m$. The mapping Ψ additionally needs the exact recording time t of each camera, cf. Sect. 4.

3.2 Navigation space partitioning

The embedding Ψ results in a three-dimensional point cloud, each point (φ, θ, t) representing a sample of the plenoptic function. In order to reconstruct an arbitrary virtual view from this sample set, we need a partition of the 3D space to efficiently fetch the closest samples. For a d-dimensional space, Chen and Williams proposed to generate some arbitrary graph to partition the space such that every possible viewpoint lies in a d-simplex and can be expressed as a linear combination of the $d+1$ vertices of the enclosing simplex [4].We extend this idea and apply a constrained Delaunay tetrahedralization to the set of navigation space points. The tessellation is constrained such that every tetrahedron consists of at most three vertices with different φ and θ, i.e., three different real cameras. The fourth vertex always represent an image of one of the other three cameras shifted along the temporal axis. This is necessary to avoid artifacts during rendering since cameras only map approximately to the polynomial surface due to the least-squares approach.

We want to point out that our navigation space $\mathcal{N}$ is not equivalent to 4D space-time. Our virtual viewpoint is spatially restricted: we can viewpoint-navigate on the hull spanned by all camera recording positions, looking at the scene from different directions, but we cannot, for example, move into the scene or fly through the scene.

3.3 Image interpolation in space and time

With the subdivision of the navigation space $\mathcal{N}$ into tetrahedra, each point $\mathbf{v}$ is defined by the vertices of the enclosing tetrahedron $\lambda = \{\mathbf{v}_i\}, i = 1 \ldots 4$. Its position can be uniquely expressed as $\mathbf{v} = \sum_{i=1}^{4} \mu_i \mathbf{v}_i$, where μ_i are the barycentric coordinates of $\mathbf{v}$. Each of the 4 vertices $\mathbf{v}_i$ of the tetrahedron corresponds to a recorded image I_i, each of the 12 edges e_{ij} correspond to a bidirectional correspondence map $\mathbf{W_{ij}}$ which is estimated in a preprocessing stage, cf. Sect. 4. We are now able to synthesize a novel image I_v for every point $\mathbf{v}$ inside the recording hull of the navigation space $\mathcal{N}$ by multi-image interpolation:

$$I_v = \sum_{i=1}^{4} \mu_i \tilde{I}_i,$$

where

$$\tilde{I}_i \left(\pi_i(\mathbf{x}) + \sum_{j=1,\ldots,4, j \neq i} \mu_j \pi_j(\mathbf{W_{ij}}(\mathbf{x})) \right) = I_i(\mathbf{x})$$

are the forward-warped images [11]. $\{\pi_i\}$ defines a set of re-projection matrices π_i that map each image I_i onto the image plane of I_v, as proposed by Seitz and Dyer [15]. Those matrices can be easily derived from camera calibration, cf. Sect. 4. Image re-projection is done on the GPU without image data resampling. We handle occlusion on-the-fly based on depth heuristics as proposed by Stich et al. [17]. Disoccluded regions are detected measuring the triangle stretch of the underlying warping mesh. In those regions, the mesh is cut open using a geometry shader. As a last step, the borders of the rendered images are cropped, since no reliable correspondence information is available for these regions.

4. ACQUISITION AND PRE-PROCESSING

Having laid out the theoretical foundations for our space-time image interpolation algorithm, we now explain how we pre-process multi-view footage for use in our STF/X editor.

Scene capture. To acquire multi-video data, we use a sparse setup between 5 to 16 HDV Canon XHA1 camcorders (1440 x 1080 pixels, 25 fps), Fig. 1. The cameras run completely on batteries and can be used indoors and outdoors.

Figure 1: *Recording setup: for our STF/X editor, consumer-grade camcorders suffice. The setup is flexible: adjacent cameras are only required to have sufficient view overlap and the angle between viewing directions should not exceed 10 degrees.*

Color correction. To correct for color balance differences between cameras, we use the Master camera $\mathbf{c}_m$defined in the embedding stage, Sect. 3.1, and apply to all video frames the color correction approach presented by Snavely et al. [16].

Camera calibration. We need to determine $\mathbf{R}$ and $\mathbf{p}$ of the camera setup to define the mapping Ψ from world space coordinates to navigation space $\mathcal{N}$, Sect. 3.1. Recent structure-from-motion algorithms for unordered image collections [16] solve this problem robustly in an offline process. They also provide a set of sparse world space points suitable for constructing the common ground plane for our navigation space. We found that this algorithm yields robust results also for dynamic scenes.

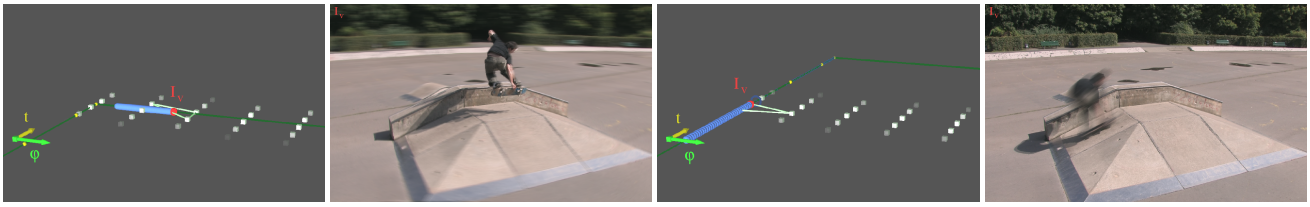
Camera synchronization. For the mapping Ψ additionally the exact recording time t of each camera must be known. Since our cameras are not gen-locked, we estimate the sub-frame temporal offset using an image-based approach proposed by Meyer et al. [13].

Dense correspondence field estimation. In order to interpolate between two images I_i, I_j, we need bidirectional dense correspondence maps $\mathbf{W_{ij}}$ for each tetrahedral edge in navigation space $\mathcal{N}$, Sect. 3.1. In our STF/X editor, we employ the algorithm proposed by Stich et al. for dense correspondence estimation [18]. For most image pairs, the results are perceptually convincing; if not, the algorithm accepts manual corrections. Automatic correspondence estimation typically takes around 2 minutes per image pair.

Object masking. For some visual effects described in Sect. 5, we need binary masks of foreground objects. At the current stage, we use the algorithm of Bai et al. [1] for this purpose.

5. THE STF/X EDITOR

The second major contribution of our paper is the STF/X editor. On the website of Digital Air Inc. [5] a list of visual effects are described that can be produced with specific camera setups and different exposure techniques. We distinguish between effects that are solely based on camera placement and different triggering schemes (*frozen moment, start-stop, slow motion, time ramps, space ramps and match cut*) and effects that rely on a combination of camera placement and exposure techniques (*space blur, time blur, multi-exposure, flash trail, open flash and progressive motion distortion*)[20]. In the following, we show that these effects can be realized by our STF/X editor from a single recording alone. In contrast to contemporary space-time visual effects production, our approach is entirely a post-production process. We record a scene once with up to 16 cameras; afterwards, we can create various space-time visual effects from this single capture as opposed to designing and capturing each single effect separately.



(a) Space blur sampling and resulting blurred scene background.

(b) Time blur sampling and resulting blurred skater.

Figure 2: *Different space-time sampling examples and resulting images. The sampling pattern is visualized in our navigation space $\mathcal{N}$: the white cubes represent recorded plenoptic samples, the red dot is the current virtual view I_v, the blue dots visualize the additional samples.*

5.1 Path-based effects

In [20], Wolf describes space-time visual effects in two-dimensional space-time diagrams, allowing for intuitive visualization and planning of visual effects. In his representation, each effect maps to a specific path within the diagram. Our navigation space, cf. Sect. 3, is the three-dimensional counterpart of Wolf's space-time diagrams which can be interactively explored. Within this space, arbitrary camera paths are defined by placing, editing, or deleting control points making use of common 3D animation concepts. The effect camera path through space-time is interpolated by Catmull-Rom splines.This way, various effects can be generated by designing an appropriate camera path: *frozen moments* map to paths with constant time, *slow motion* and *time/space ramps* are created by varying the sampling density along the path, and *match cuts* amount to jumps to specific points in space-time.

5.2 Exposure effects

Visual effects based on unconventional exposure settings are very common in still photography. The longer the shutter is open, the more light is captured over time, and moving objects appear blurred. Long exposures are also often combined with flash lights to accentuate one or more specific moments. In video production, however, these effects are created in post-production due to physical limitations of the longest possible exposure time. Instead of taking a single long exposure, the scene is typically captured with high frame-rate cameras with short exposure times, and the long exposure is synthesized via image accumulation.

Simulating an open shutter. To simulate an open camera shutter in our STF/X editor, we integrate along a parametric path $p(s)$, $s \in [0,1]$, in our navigation space $\mathcal{N}$ by accumulating virtual views. The resulting image is

$$\mathbf{I}_O = \int_{p(s)} I_v(p(s))\mathrm{d}s, \tag{1}$$

where $p(s)$ defines a sample position in $(\varphi,\theta,t) \in \mathcal{N}$. Suitably discretized, Eq. (1) serves as basis for various shutter effects.

Space blur, time blur and multi-exposure. Motion blur is the most prominent effect of long exposure times and has been thoroughly investigated, see [6] for a good discussion. In our STF/X editor, we adapt the accumulation approach by Haeberli et al. [7] since it seamlessly integrates with flash effects described below.

Depending on the desired effect, i.e., blur in time, space or both, different sampling paths $p(s)$ have to be used to solve the integral in Eq. (1). For pure time blur, the discretized and weighted version of Eq. (1) reads

$$I_O = \sum_{n=0}^{N-1} w(n) \cdot I_v(\varphi,\theta,t_0 + n \cdot \frac{(t_1 - t_0)}{N-1}), \tag{2}$$

with the weighting term $w(n), \sum_{n=0}^{N-1} w(n) = 1$. N describes the number of samples used in this temporal supersampling strategy and steers the quality of the long exposure, with more samples resulting in higher quality. Sampling along one or both spatial axes of the navigation space $\mathcal{N}$ while keeping the time constant will result in space blur. Fig. 2 shows different sampling patterns in space-time for time and space blur along with the resulting image. Using a small N results in a composition of N distinct moments in space-time within one frame, called multi-exposure, Fig. 4(a).

Flash effects. Combining long exposures with a single or multiple flash strobes, *open flash* and *flash trail* effects are generated. Again, using the accumulation technique of Eq. (2), these effects are generated by combining a long exposure with one or more short exposure renderings taken at the respective point in space-time, Fig. 4(b). The weights are adjusted such that higher weight is given to the flash samples.

5.3 Advanced shutter effects

Our approach is not restricted to effects based on exposure techniques. We are also able to simulate arbitrary - and physically impossible - shutter timings. Consider for example a rolling shutter: such a shutter often results in a progressive distortion of a moving object where each scanline depicts a slightly different instant of time, Fig 3(a). To emulate this effect in our STF/X editor, the rendered special effect is composed of scan lines taken from different samples along the temporal axis in space-time. We are not restricted to sampling only along the temporal axis; sampling along the spatial axes would result in cubistic views of an object. In principle, each pixel of the virtual view can be taken from any arbitrary position in our space-time navigation space.

5.4 Non-photorealistic effects

Our system is not limited to photorealistic effects. Since we are operating on images only, any artistic stylization that can be applied to a single image or a video stream naturally maps to our STF/X editor as well. We extend the algorithms for motion stylization proposed by Kim and Essa [8] and integrate them into our STF/X editor. Motion stylization is typically applied to objects to amplify their perceived motion. In contrast to [8], we do not segment the moving object on-the-fly but use the binary masks created in Sect. 4. In order to get a mask for an arbitrary point in space-time, we apply the same rendering technique as for the virtual views described in Sect. 3.3. We then use those masks to track a set of particles along the trailing contour of the object. These particle tracks serve as basis for the rendition of particles or speedlines as shown in Figs. 3(b,c).

5.5 The effect graph

Since all of the described effects use only image data as input to render F/X images as output, it is straight-forward to combine effects. A multi-stage rendering pipeline can be setup to create even

(a) Scanline progressive motion distortion. (b) Particle effect. (c) Speedline effect.

Figure 3: *Shutter effect and motion stylization techniques.*

more complex visual effects. To define a proper rendering flow, effects are combined in a digraph structure. The root of the graph holds the effect responsible for rendering the final frame. All other vertices in the graph act as input sources for connected effects, with directed graph edges indicating the flow of image data. Effects with no input sources take the underlying STF/X editor as data source. The edges of the graph can be weighted, allowing emphasis of one effect over another. An example of such an effect combination is shown in Fig. 4(b) where we combine *space blur* with *open flash*.

6. IMPLEMENTATION DETAILS

The interpolation algorithm described in Sect. 3.3 is implemented entirely on GPU and exceeds 25 fps on an NVIDIA GeForce 8800 GTX at 960x540 pixels output resolution. Currently, the framerate is limited by the time it takes to transfer the data from secondary storage to host memory and finally to GPU memory. To further accelerate our STF/X editor, we employ a least-recently-used caching strategy [14], caching images and correspondence fields in host memory, as well as trying to keep them resident in GPU memory as long as possible.

Many of the effects described in the previous section require a larger number of space-time samples as input. While the underlying STF/X editor allows interaction in real-time, performance quickly deteriorates if many space-time samples have to be rendered over and over again. Fortunately, it is very likely that space-time samples can be reused when rendering many consecutive frames with the same visual effect. This especially holds true for effects based on accumulation. We exploit this property by again incorporating a least-recently-used caching strategy into our STF/X editor, this time caching previously used space-time samples. This way, the rendering overhead is reduced to a minimum and the real-time performance of the system is kept.

7. RESULTS AND DISCUSSION

We evaluate our STF/X editor on a variety of different real-world scenes that we captured with 5 to 16 consumer-grade camcorders, Table 1. Besides scene-specific challenges, such as high-speed motion, motion blur and over-/underexposure, our approach must also cope with lens distortion, camera noise, and compression artifacts. Fig. 4 shows still images of different effects created with our editor. The accompanying video presents a number of different STF/X sequences to help assess our approach, including a comparison to the approach of Zitnick et al. [23].

Because we employ the well-known accumulation technique to emulate camera shutter effects, we must decide on a number of plenoptic samples N to be summed up in Eq.(2). Depending on output rendering resolution, GPU speed, and magnitude of scene motion, the number of samples is kept adaptive to ensure high-quality results without compromising performance. For the test scenes shown in the accompanying video, we have used $N = 25$ to obtain high-quality long-exposure effects.

Scene	**Cameras**	**Effects**	**Additional samples per frame**
Firebreather	16	time& space ramps freeze-rotate & slow motion	0
Skateboard	6	open flash & space blur freeze-rotate	5 + 25
Yuki	5	progressive motion distortion	121
Freerunner	16	time/space blur & multi-exposure freeze-rotate & slow motion	25 + 20 + 3
Skateboard	6	temporal flare freeze-rotate	0

Table 1: *Effect configuration details for our test scenes, Fig. 4. The number of additional samples for each effect is given in the last column. Recall that despite the high number of additional space-time samples needed for some effects, our STF/X editor still exceeds 25 fps.*

To ensure high-quality STF/X results, we observed that the angle between adjacent camcorders should not exceed about 10 degrees, independent of scene content. As a rule of thumb, we found that across space or time, scene correspondences should not be separated by more than approximately 10% of linear image size for contemporary matching algorithms to achieve satisfactory results. Despite this restriction on camera viewpoints, a large view range can be covered with a single setup since the cameras can be placed freely. The total pre-processing time for the steps described in Sect. 4 for a typical scene as shown in the accompanying video amounts to 1-2 hours, covering color correction, camera calibration and synchronization, and object masking. Correspondence estimation takes another 2 hours on a single PC. This process, however, is completely automatic.

Our STF/X editor does not rely on any specific correspondence finding algorithm. We found that the automatic, pair-wise image-correspondence estimation algorithm by Stich et al. [18] yields convincing and robust results. Remaining small inaccuracies in the correspondence fields can be corrected by user interaction. Correspondence correction typically takes about another hour per scene.

(a) Multi-exposure. (b) Open flash used as input for space blur. (c) Space and time ramps.

Figure 4: *Stills from the examples shown in the accompanying video. The input data has been recorded with 5-16 Canon XHA1 camcorders. Each scene shows different visual effects.*

8. CONCLUSION

We have presented a space-time visual effects editor to create and interactively edit visual effects as a post-production process. Based on refined image interpolation, our STF/X editor supports various space-time visual effects. Our approach allows creating space-time visual effects from sparse, unsynchronized multi-video footage of arbitrary, dynamic real-world scenes. We do not need special acquisition hardware or time-consuming setup procedures. Instead, we record with consumer camcorders in arbitrary environments. A single multi-video recording suffices to generate arbitrary effects, providing high flexibility and a considerable reduction in time and effort for STF/X production.

Future work will focus on adding high-level editing operations to our STF/X editor, like altering scene illumination and object appearance. By exploiting all inter-relationships between multi-view video frames, correspondence estimation will be further improved. Highly reliable dense correspondences will enable us to propagate image editing operators automatically to the entire set of multi-video images.

9. REFERENCES

[1] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video SnapCut: robust video object cutout using localized classifiers. *ACM Transactions on Graphics*, 28(3):70:1–70:11, 2009.

[2] T. Beier and S. Neely. Feature-based image metamorphosis. In *Computer Graphics (Proceedings of SIGGRAPH 92)*, volume 26, pages 35–42, New York, 1992. ACM.

[3] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured Lumigraph Rendering. In *Proceedings of SIGGRAPH 2001*, pages 425–432, New York, 2001. ACM Press/ACM SIGGRAPH.

[4] S. E. Chen and L. Williams. View interpolation for image synthesis. In *Proceedings of SIGGRAPH 1993*, pages 279–288, New York, 1993. ACM Press/ACM SIGGRAPH.

[5] Digital Air Inc. Digital Air Techniques. http://www.digitalair.com/techniques/index.html, 2009.

[6] K. Egan, Y.-T. Tseng, N. Holzschuch, F. Durand, and R. Ramamoorthi. Frequency analysis and sheared reconstruction for rendering motion blur. *ACM Transactions on Graphics*, 28(3):93:1–93:13, 2009.

[7] P. Haeberli and K. Akeley. The accumulation buffer: hardware support for high-quality rendering. In *Computer Graphics (Proceedings of SIGGRAPH 90)*, pages 309–318, New York, NY, USA, 1990. ACM.

[8] B. Kim and I. Essa. Video-based nonphotorealistic and expressive illustration of motion. In *Proc. CGI '05*, pages 32–35, Washington, DC, USA, 2005. IEEE Computer Society.

[9] S. Lee, G. Wolberg, and S. Shin. Polymorph: morphing among multiple images. *IEEE Computer Graphics and Applications*, pages 58–71, 1998.

[10] D. Mahajan, F. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur. Moving Gradients: A Path-Based Method for Plausible Image Interpolation. *ACM Transactions on Graphics*, 28(3):42:1–42:11, 2009.

[11] W. Mark, L. McMillan, and G. Bishop. Post-Rendering 3D Warping. In *Proc. of Symposium on Interactive 3D Graphics*, pages 7–16, 1997.

[12] L. McMillan and G. Bishop. Plenoptic Modeling. In *Proceedings of SIGGRAPH 1995*, pages 39–46, New York, 1995. ACM Press/ACM SIGGRAPH.

[13] B. Meyer, T. Stich, M. Magnor, and M. Pollefeys. Subframe Temporal Alignment of Non-Stationary Cameras. In *Proc. of BMVC '08*, pages 103–112, 2008.

[14] E. J. O'Neil, P. E. O'Neil, and G. Weikum. The LRU-K page replacement algorithm for database disk buffering. *SIGMOD Rec.*, 22(2):297–306, 1993.

[15] S. M. Seitz and C. R. Dyer. View Morphing. In *Proceedings of SIGGRAPH 1996*, pages 21–30, New York, 1996. ACM Press/ACM SIGGRAPH.

[16] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics*, 25(3):835–846, 2006.

[17] T. Stich, C. Linz, G. Albuquerque, and M. Magnor. View and Time Interpolation in Image Space. *Computer Graphics Forum (Proc. of PG'08)*, 27(7):1781–1787, 2008.

[18] T. Stich, C. Linz, C. Wallraven, D. Cunningham, and M. Magnor. Perception-motivated Interpolation of Image Sequences. In *Proc. of ACM APGV'08*, pages 97–106, Los Angeles, USA, 2008. ACM Press.

[19] H. Wang, M. Sun, and R. Yang. Space-Time Light Field Rendering. *IEEE Trans. Visualization and Computer Graphics*, pages 697–710, 2007.

[20] M. Wolf. Space, Time, Frame, Cinema: Exploring the Possibilities of Spatiotemporal Effects. *New Review of Film and Television Studies*, pages 369–374, Dec. 2006.

[21] Z. Zhang, L. Wang, B. Guo, and H.-Y. Shum. Feature-based light field morphing. In *Proceedings of SIGGRAPH 2002*, volume 21, pages 457–464, New York, 2002. ACM.

[22] K. C. Zheng, A. Colburn, A. Agarwala, M. Agrawala, D. Salesin, B. Curless, and M. F. Cohen. Parallax photography: creating 3d cinematic effects from stills. In *Proc. GI '09*, pages 111–118, 2009.

[23] C. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-Quality Video View Interpolation Using a Layered Representation. *ACM Transactions on Graphics*, 23(3):600–608, 2004.