Interactive Scene Flow Editing for Improved Image-based Rendering and Virtual Spacetime Navigation

Kai Ruhl[1], Martin Eisemann[1,2], Anna Hilsmann[3], Peter Eisert[3] and Marcus Magnor[1]
[1]TU Braunschweig, Germany [2]FH Koeln, Germany [3]Fraunhofer HHI, Berlin, Germany
{ruhl,magnor}@cg.cs.tu-bs.de, martin.eisemann@fh-koeln.de, {anna.hilsmann,peter.eisert}@hhi.fraunhofer.de *

ABSTRACT

High-quality stereo and optical flow maps are essential for a multitude of tasks in visual media production, e.g. virtual camera navigation, disparity adaptation or scene editing. Rather than estimating stereo and optical flow separately, scene flow is a valid alternative since it combines both spatial and temporal information and recently surpassed the former two in terms of accuracy. However, since automated scene flow estimation is non-accurate in a number of situations, resulting rendering artifacts have to be corrected manually in each output frame, an elaborate and time-consuming task. We propose a novel workflow to edit the scene flow itself, catching the problem at its source and yielding a more flexible instrument for further processing. By integrating user edits in early stages of the optimization, we allow the use of approximate scribbles instead of accurate editing, thereby reducing interaction times. Our results show that editing the scene flow improves the quality of visual results considerably while requiring vastly less editing effort.

Categories and Subject Descriptors

I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*Depth cues*; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*Motion*; H.5.2 [**Information Interfaces and Representation**]: User Interfaces—*Interaction styles*

Keywords

scene flow; stereo; optical flow; interactivity; user interface; image-based rendering; view interpolation

1. INTRODUCTION

In visual media productions, several editing operations in post-production require information about scene depth and/or motion. Due to the current popularity of stereoscopic 3D movies, depth has taken a prominent role for all manner of stereo post production tasks [21], while motion is used primarily for slow-motion, frame upsampling and motion-based effects such as motion blur.

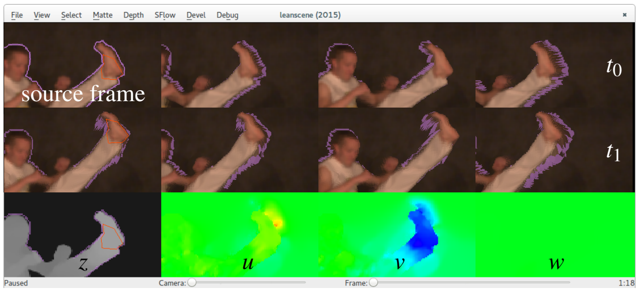


Figure 1: User interface for a scene recorded by four cameras. Top and middle row: View interpolation towards the cameras at times t_0 and t_1 using one source frame and the current scene flow. The user can toggle the warped vs. recorded frames to identify visual artifacts. Bottom row: color-coded visualization of the current depth z and 3D motion u, v, w estimate.

Up to now, multimedia authoring tools have estimated depth and motion separately using only two images each, an under-determined and ill-posed problem. Since there are strong links between spatial and temporal image correspondences, i.e. object texture and boundaries are present in both, using joint optimization in the form of scene flow [15] is a promising direction to improve the robustness of the estimation; stereo and optical flow are then mere reprojections of the scene flow into specific cameras at specific times.

Surprisingly, for over a decade scene flow has not been able to reach the quality of dedicated stereo and optical flow methods. Only recently, scene flow algorithms have begun to outperform the former two [16]. This gives rise to the idea of multimedia content production tools based on scene flow.

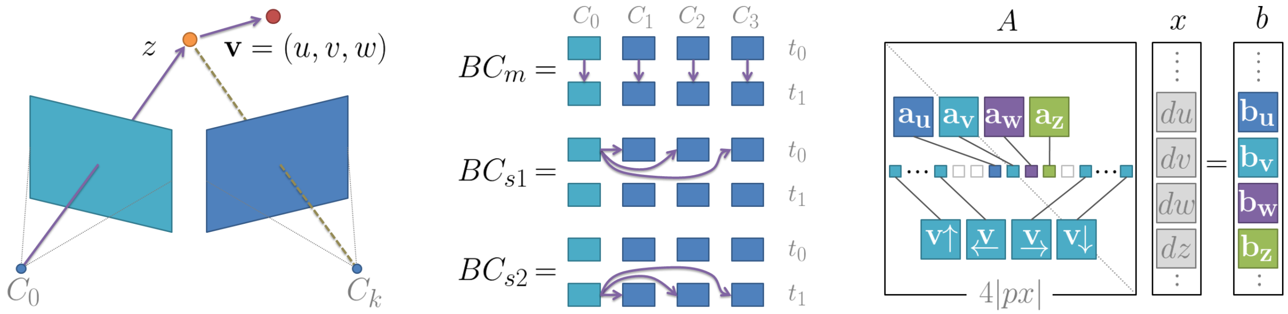
However, similar to stereo/optical flow, scene flow algorithms are not perfect, showing typical failure cases e.g. for repeating structures, occlusions and violations of the color constancy assumption. Flow field artifacts manifest themselves as visual artifacts in rendered output frames, e.g. for virtual camera views. Usually, visual artists employ image editing tools such as Adobe Photoshop to repair those visual artifacts frame-by-frame. Alternatively, they can use keyframe animation in tools such as TheFoundry NUKE or Adobe AfterEffects to model spatiotemporal transitions manually. Both are elaborate and time-consuming tasks whose effort is linear in the number of output frames or transitions.

For this reason, first editing tools for stereo and optical flow fields have recently been developed. They range from relatively direct cut&paste tools [6] to shape-fitting approaches [23]. However, to the best of our knowledge, no existing multimedia authoring tool includes scene flow yet. Our contribution is the first workflow that provides scene flow editing capabilities, allowing *interactive* manipulation of an *ongoing* scene flow estimation.

*Video and further paper details available under:
`http://graphics.tu-bs.de/publications/ruhl2015acmmm/`

MM'15, October 26 - 30, 2015, Brisbane, Australia

ACM 978-1-4503-3459-4/15/10...$15.00
DOI: `http://dx.doi.org/10.1145/2733373.2806251`



(a) scene flow variables **(b) data term components** **(c) inner $A\mathbf{x} = \mathbf{b}$ equations; some row for v**

Figure 2: Optimization details. (a) The 4D flow components z, u, v, w are modeled in world space, allowing for an arbitrary number of cameras to support the "hero camera" C_0; arrows denote depth and motion. (b) The data term comprises a motion term $BC_m(z, \mathbf{v})$, a depth term $BC_{s1}(z)$ and a depth-after-motion term $BC_{s2}(z, \mathbf{v})$; arrows denote the frames used for data term evaluation. (c) The $A\mathbf{x} = \mathbf{b}$ system of linear equations links u, v, w and z and retrieves support information from the 4-neighborhood, here: a row for v.

1.1 Related Work

In multimedia authoring, depth map editing has become popular with the latest recurrence of stereoscopic 3D movies, having two complementary approaches: stereo conversion for videos shot with a single camera, and stereo estimation for footage recorded by two cameras. In stereo conversion, 2D video is converted to 3D video by manually creating a depth map for each input frame. Current approaches use scribble-based interfaces to draw depth scribbles and interpolate for the remaining pixels [5, 18], or use a set of sparse depth (in)equalities to add depth to cartoons [14]. For footage captured with stereo cameras, user interaction is often used to guide stereo matching. Current approaches provide sparse ground truth initializations in the form of point correspondences [17] or matching splines [9] while we use matching regions; remove outliers for better depth interpolation [3] or restrict the cost volume and enhance local depth resolution [10]; use geometric model fitting and discontinuity brushes in a belief propagation framework [23] where we use discontinuity scribbles; or modify local weights in a variational energy functional [4], which is part of our approach too.

Optical flow editing is useful for all manner of temporal effects, or when employing multiple non-synchronized cameras [7]. Current approaches use cut&paste on a flow field to match regions via perspective transformation and to recompute optical flow locally [6], or provide approximate correspondence regions which are then refined within further optimization [11], similar to our method.

To the best of our knowledge, no approach for *scene flow editing* exists. The closest in spirit to our method are the *stereo editing* approach by Doron et al. [4] and the *optical flow editing* approach by Ruhl et al. [11] in the sense that both modify a variational energy functional, and the latter exploits the image pyramid for refining approximate user input.

1.2 User Interface

Our objective is the creation of smooth spatiotemporal view interpolation. Fig. 1 shows our user interface: A scene has been recorded with four cameras C_k at two time steps t_0 and t_1, yielding frames I_t^k. One source frame has been selected (I_0^0 in the following, w.l.o.g.) and warped fully towards all other cameras and time steps using the scene flow. Ideally, the interpolated frames are equal to the recorded frames I_t^k; the user can check this by toggling between recorded/interpolated frames.

The scene flow is estimated in the background using a coarse-to-fine image pyramid with levels $L = L_{\max}..0$, where 0 is the finest resolution level. At each step of the optimization, the user interface is updated to show the warped frames at increasingly higher resolutions and warped using better scene flow estimates. The user observes the optimization and identifies visual artifacts in the warped frames as soon as they become apparent on some (early) level L. Once such an error has been identified, the user pauses the estimation and uses our editing operations to guide the algorithm.

This guidance is taken as coarse initialization or soft constraint while the ongoing optimization determines the subpixel-precise final solution. In this manner, we benefit from both human scene recognition and subsequent algorithmic refinement.

2. PROBLEM FORMULATION

In order to develop appropriate editing operations, we first have to analyze the failure modes of scene flow estimation. We base our analysis on the well-known multi-view scene flow by Basha et al. [1, 8]. Scenes are recorded with an arbitrary number K of cameras $C_{0..K-1}$, one of which is designated as the so-called "hero" camera, at two frames t_0 and t_1, making it usable not only for stereo cameras but also e.g. for trifocal cameras [12]. For the hero camera (C_0 in the following, w.l.o.g.), we reconstruct the per-pixel depth z and the 3D motion $\mathbf{v} = (u, v, w)$ in world space, Fig. 2 (a). Assuming brightness constancy, we use a variational energy minimization:

$$E(z, \mathbf{v}) = \int_\Omega \underbrace{(BC_m + BC_{s1} + BC_{s2})}_{\text{data term}} + \alpha \underbrace{(S_m + \mu S_s)}_{\text{smoothness term}} \, dxdy \qquad (1)$$

with α balancing data vs. smoothness and μ balancing motion vs. depth smoothness. Using recorded frames I_t^k from cameras C_k at time steps $t = 0..1$, the subterms are, Fig. 2 (b):

$$BC_m(z, \mathbf{v}) = \sum_{k=0}^{K-1} o_m^k \cdot \psi(|I_0^k(\mathbf{p}_0^k) - I_1^k(\mathbf{p}_1^k)|^2) \qquad (2)$$

$$BC_{s1}(z) = \sum_{k=1}^{K-1} o_{s1}^k \cdot \psi(|I_0^0(\mathbf{p}_0^0) - I_0^k(\mathbf{p}_0^k)|^2) \qquad (3)$$

$$BC_{s2}(z, \mathbf{v}) = \sum_{k=1}^{K-1} o_{s2}^k \cdot \psi(|I_1^0(\mathbf{p}_1^0) - I_1^k(\mathbf{p}_1^k)|^2) \qquad (4)$$

$$S_m(\mathbf{v}) = \psi(|\nabla u|^2 + |\nabla v|^2 + |\nabla w|^2) \qquad (5)$$

$$S_s(z) = \psi(|\nabla z|^2) \qquad (6)$$

with $BC_{m,s1,s2}$ the brightness constancy data terms for motion, depth and depth-after motion, $S_{m,s}$ the smoothness terms for mo-

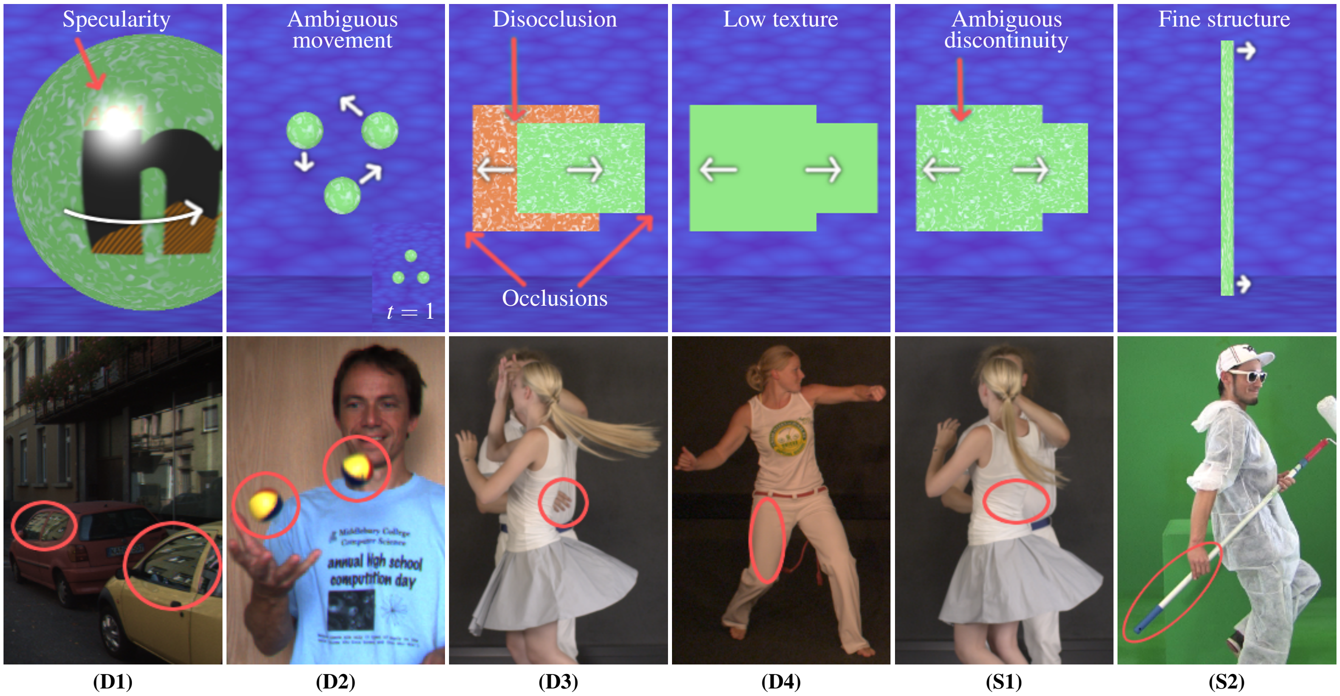


Figure 3: Common artifact causes. (D1) color constancy violation: the ball rotates to the right while the specularity stays in place. (D2) ambiguous displacements of three spheres (D3) disoccluded region without proper source region (D4) low-textured regions (S1) ambiguous discontinuities: not all image gradients are discontinuities. (S2) fine-scale objects. See Fig. 4–7 for artifact details.

tion and depth; occlusion maps $o_{m,s1,s2}$ deactivating the data term locally, image space points $\mathbf{p}^k_{t=0}(z)$ and $\mathbf{p}^k_{t=1}(z,\mathbf{v})$ reprojected from world space points $\mathbf{P}_0$ and $\mathbf{P}_1$ into a camera k, and the non-quadratic robust Charbonnier penalty $\psi(s^2) = \sqrt{s^2+\varepsilon^2}$ [13].

Now where does this model have failure cases? We identified six typical situations, exemplified in Fig. 3 by both synthetic and real-world scenes. Row 1 shows input frames leading to artifacts; row 2 additionally motivates the scenarios. Four problems arise from the data term **(D1–4)** and two from the smoothness term **(S1–2)**:

(**D1**) Violations of brightness or color constancy assumption. Any reflectance, specularity or other non-Lambertian property such as shadows can cause pixels in one image to simply not look the same in other images, invalidating the data terms $BC_{s1,s2}$ spatially and/or BC_m temporally. Thus, we would like to deactivate the data term locally, leaving the region to the smoothness term.

(**D2**) Spatial ambiguities or large displacements. Repeating or similar structures have multiple local minima and can be incorrectly matched in the energy minimization. Also, since variational approaches such as ours handle large displacements on coarse image pyramid levels, an object smaller than its motion between t_0 and t_1 might be oversmoothed in the pyramid during downsampling, effectively vanishing. In these cases, we would like to give the algorithm some correspondence hint.

(**D3**) Occluded regions. Between two images I^k_t, occluded pixels have no proper target in $BC_{m,s1,s2}$, violating the color constancy assumption. Other images might still contain the same region, so in this case we would like a highly selective version of data term deactivation. Additionally, should no image contain the occluded region, smoothness will now fill the region from all sides, which is usually not the desired effect, so we would like to be able to influence the smoothness propagation direction.

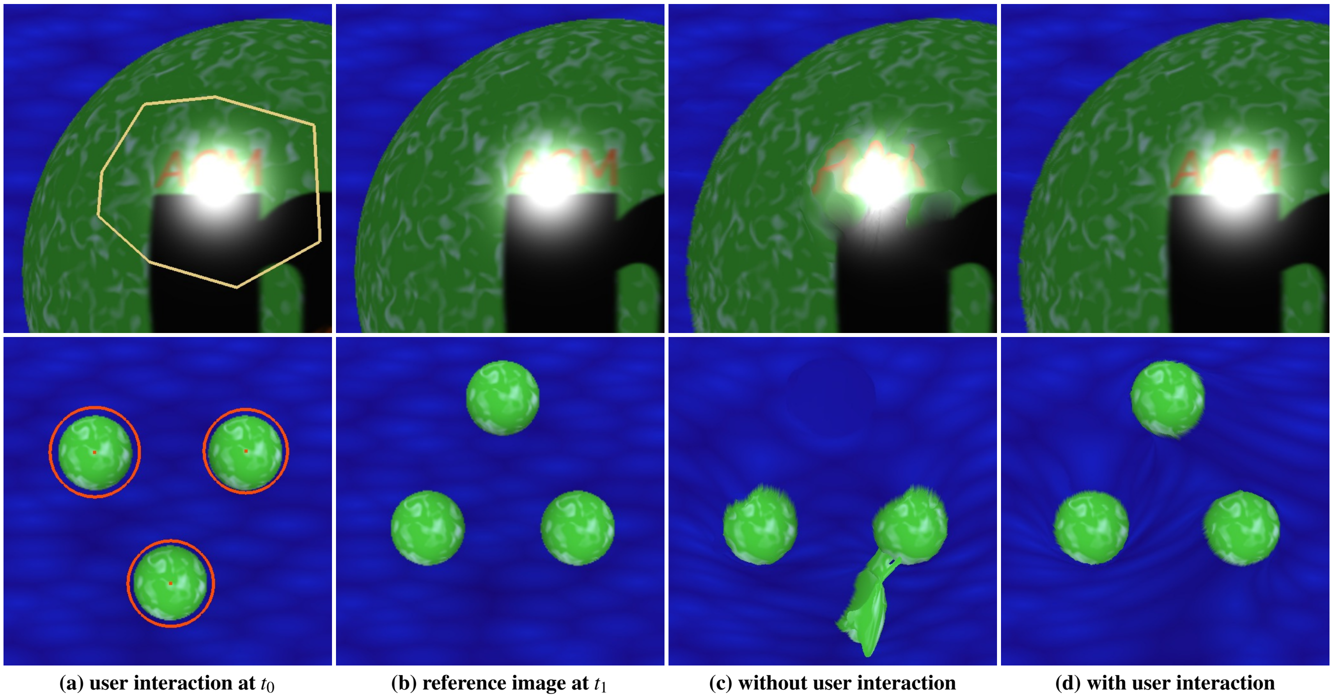
(**D4**) Low-textured regions. A uniform case of ambiguous matching, low texture is ideally resolved by the smoothness terms $S_{m,s}$, since the data term has equal penalties everywhere. However, noise has a comparatively large influence in $BC_{m,s1,s2}$. Thus, we would like to either provide some hint regarding matching regions, or promote uniform z and $\mathbf{v}$ for that region.

(**S1**) Discontinuities. The smoothness terms $S_{m,s}$ do not actively detect object boundaries, and uniformely demand e.g. $|\nabla u|$ in Eq. 5 to be low. Robust approaches like ours avoid over-penalization of discontinuities when compared to quadratic terms but still require smoothness everywhere. Furthermore, Vogel at al. [16] argue that w motion around discontinuities can often be "simulated" by u and v motion to avoid a smoothness penalty, circumventing correct discontinuity formation. Heuristics like anisotropic regularization [20] explicitly encourage the formation of discontinuities along image gradients, but cannot differentiate between true object boundaries and texture gradients (e.g. striped shirt). Thus, we would like to specify "true" discontinuities manually, either as sharp boundary or as broad brush indicating boundary candidates.

(**S2**) Inappropriate smoothness weight. Weighting the smoothness term vs. the data term is usually done globally, here with α and μ. Depending on the image content, applying different weights to different regions may be more appropriate. Setting one global weight too high results in oversmoothing, preventing deformations and glueing objects together. Setting the weight too low results in non-smooth objects distorted by ambiguities in the data term. Therefore, we would like to control the smoothness weights α and μ per image region.

3. APPROACH

We distill the above analysis into four interactive editing tools that can be used in conjunction with each other: An edge tool (**ET**) in Sec. 3.1, an occlusion tool (**OT**) in Sec. 3.2, a smoothness


(a) user interaction at t_0

(b) reference image at t_1

(c) without user interaction

(d) with user interaction

Figure 4: Avoiding artifacts in warped images. Row 1: Correcting a brightness constancy violation (D1). The specular region is marked as "occluded", deactivating the data term (a). Warped images should look like the reference (b). Automated scene flow produces distorted results around the specularity (c). User interaction preserves the sphere's shape (d). Row 2: Correcting ambiguous matching (D2). Circular matches provide a local initialization (a). Compared to the reference (b), automated scene flow cannot determine which ball belongs to which (c). The user's local initialization is enough to converge to the correct result (d).

tool (**ST**) in Sec. 3.3, and a match tool (**MT**) in Sec. 3.4. We ported the scene flow by Basha et al. [1] to the GPU, yielding a speedup factor of 3–5 and enabling interactive feedback to the user. Our video[1] shows all presented tools in action.

To integrate the tools into the scene flow estimation, we need to consider the optimization algorithm, detailed in Sec. 2.3 of [1]. It couples the multi-resolution levels with two nested fixed-point iterations, where outer iterations update z and $\mathbf{v}$ and re-warp the images accordingly, while inner iterations compute small increments dz and $\mathbf{dv} = (du, dv, dw)$. At each inner fixed-point iteration, the Euler-Lagrange equations for the variables z, u, v and w are solved by posing an $A\mathbf{x} = \mathbf{b}$ problem, Fig. 2 (c). A is sparse and has 4 times the number of pixels rows and columns, with elements close to the diagonal referencing the other variables (e.g. v references u, w and z at the same pixel) and receiving support information from the 4-neighborhood.

3.1 Edge Tool (ET)

Human scene understanding can easily distinguish true discontinuities from in-object gradients (e.g. recognizing a striped shirt as such), so we allow the user to draw a polygon using the mouse to form an undirected edge scribble on object discontinuities in the hero camera image I_0^0, Fig 5– 7. Pixel neighborhood across such a scribble will be ignored in the smoothness term, addressing problems (**S1**) and (**D3**). This unfortunately requires precise user input. Ideally, a broad stroke could be used to define a region in which anisotropic filtering would find the exact edge location, however in practice failure cases occur mostly around less visible discontinuities (note that e.g. in [4], Fig. 2 (b) vs. (h), the discontinuity brush on the desk edge is rather ineffective). Therefore, we keep using exact scribbles.

For integration into the scene flow, we cannot modify the energy functional directly since α is applied omnidirectionally, while we desire a discontinuity perpendicular to the user defined edge. Instead, consider the realization of S_m, Eq. 5, into the neighborhood coefficients $v_\leftarrow$, $v_\rightarrow$, $v_\uparrow$ and $v_\downarrow$ in Fig. 2 (c). For a pixel $\mathbf{p} = [x, y]^T$, e.g. $v_\leftarrow$ is defined as (derived from Eq. 35 in Appendix C of [1]):

$$v_\leftarrow = -\alpha \cdot \mu_v \cdot \frac{1}{2}(\mathrm{div}^{uvw}(x,y) + \mathrm{div}^{uvw}(x-1,y)) \qquad (7)$$

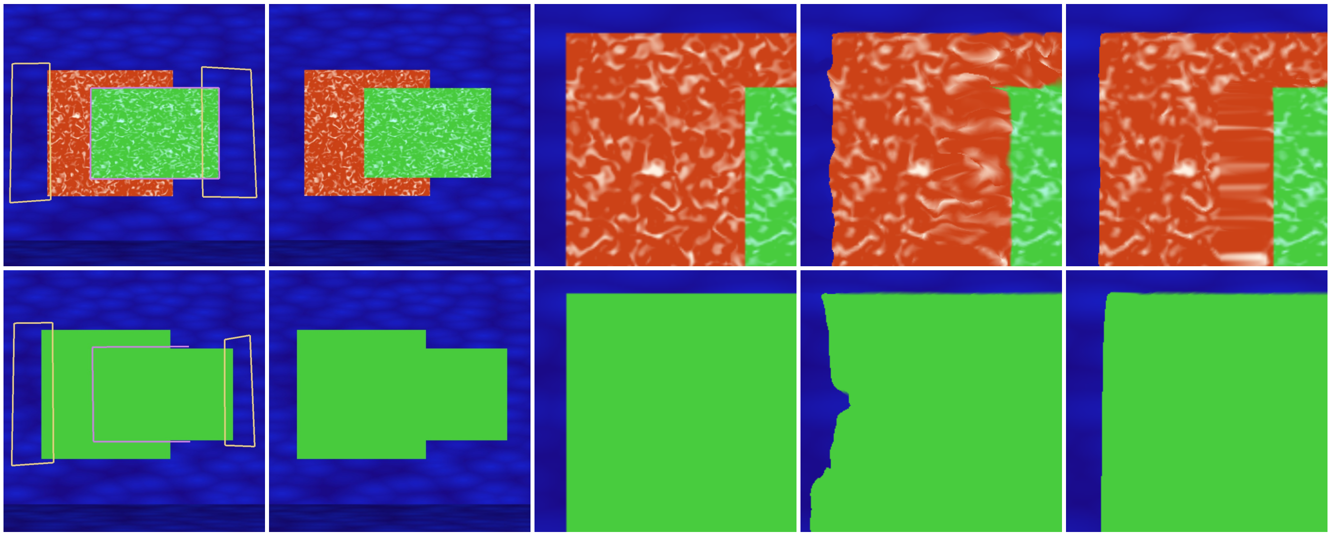
with div^{uvw} the divergence coefficients for u, v and w derived from S_m (div^z is used analoguously for realizing S_s, Eq. 6, into $z_\leftarrow$). In order to deactivate the neighborhood relation, we test whether a segment of the edge scribble intersects the line between the center points of the current pixel and the left neighbor pixel, and if so, set $v_\leftarrow$ to zero. The user can also enter a small numerical weight using the keyboard to produce a less sharply pronounced discontinuity.

3.2 Occlusion Tool (OT)

Within a mouse-clicked closed polygon on the source image I_0^0, the user can deactivate the data term w.r.t. a number of images I_t^k by selecting those images using the mouse or keyboard, Fig 4–7, making this tool useful both for true occlusions as well as for color constancy violations, addressing (**D1**) and (**D3**). As long as other cameras can see the region, the scribble can be defined very approximately. For a complete data term deactivation, the region can also be approximately defined but expected smoothness propagation must be considered; in practice, an additional edge can be used to stop unwanted propagation directions.

Regarding integration into the scene flow, consider the realiza-

[1] Video and further paper details available under: http://graphics.tu-bs.de/publications/ruhl2015acmmm/



(a) user interaction at t_0 (b) reference image at t_1 (c) zoomed at t_1 (d) without user interaction (e) with user interaction

Figure 5: Warped images. Row 1: Correcting occluded regions to the left and right, and a disoccluded region in the middle (D3), all under good texturing conditions. Occluded regions are marked as such, and an edge allows the formation of a discontinuity (a). Warped images should look like the reference at t_1 (b, c). Automated scene flow produces uneven results around the discontinuity (d), whereas the corrected version pulls the flow field apart very evenly (e). Row 2: Correcting low-textured regions (D4). Same input as above, but the edge can be placed almost arbitrarily (a). Compared to the reference (b, c) and to above, automated scene flow suffers from uneven smoothness propagation (d). The edge scribble allows for a uniform smoothness propagation towards the left (e).

tion of BC_m and BC_{s2}, Eq. 2,4, into the diagonal coefficient a_v in Fig. 2 (c). For a pixel $\mathbf{p}$, a_v is defined as (derived from Eq. 35 in Appendix C of [1]):

$$a_v = \alpha \cdot \mu_v \sum_{\mathbf{q} \in N(\mathbf{p})} \frac{1}{2} (\mathrm{div}^{uvw}(\mathbf{p}) + \mathrm{div}^{uvw}(\mathbf{q})) \tag{8}$$

$$+ \sum_{k=0}^{K-1} o_m^k \psi_m^k(.) \cdot (\mathrm{Iw}_{t[v]}^k)^2 + \sum_{k=1}^{K-1} o_{s2}^k \psi_{s2}^k(.) \cdot (\mathrm{Iw}_{t[v]}^k - \mathrm{Iw}_{t[v]}^0)^2$$

with N the 4-neighborhood; ψ_m and ψ_{s2} derived from $BC_{m,s2}$; $\mathrm{Iw}_{t[v]}^k$ the relevant images (c.f. Fig. 2 (b)) warped with the current z/$\mathbf{v}$ solution and differentiated w.r.t. v; and $\mathbf{p}$ only noted where necessary to improve readability.

The occlusion variables o_m and o_{s2} are from Eq. 2 – 4 (o_{s1} is used for calculating a_z). We locally replace o_{s1} for spatial occlusions and o_m and o_{s2} for temporal occlusions, or all for total data term deactivations, e.g. for a moving specular region. When set to zero, the data terms $BC_{m,s1,s2}$ are effectively omitted and do not factor into a_v at all, leaving the smoothness $S_{m,s}$ as the only influence.

3.3 Smoothness Tool (ST)

Within a mouse-clicked closed polygon on the source image I_0^0, the user can assign stronger or weaker smoothness weights α and μ using the keyboard, Fig 7 row 2, addressing (**D4**) and (**S2**). Undersmoothing can be easily solved by selecting a region and increasing α and/or μ. Oversmoothing can be solved either by decreasing α/μ or by providing an edge scribble. In practice, a common strategy is to define regions with increased smoothness as slightly too large and then using the edge tool to encourage discontinuity formation.

For integration into the scene flow, consider a_v in Eq. 8 and $v_{\leftarrow}$ in Eq. 7 again. To modify the smoothness, we locally replace α and μ by custom user values defined within the closed smoothness scribble. It is also possible to define α_Z, α_u, α_v and α_w separately (same for μ), but in practice this is rarely needed.

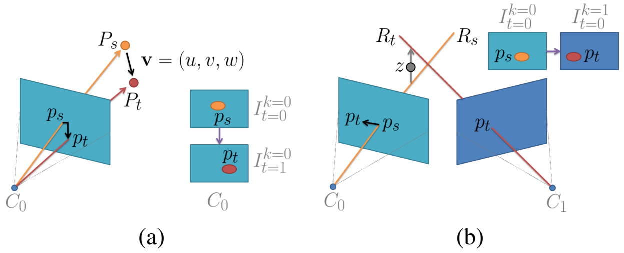


Figure 6: Temporal (a) or spatial (b) match tool (MT) details.

3.4 Match Tool (MT)

Starting with a mouse-clicked closed polygon or circle in the source image, the user can achieve an approximate target displacement into another image either directly by using the mouse to click the new location or by using the keyboard to apply translation, rotation, and scaling, Fig. 4 row 2, similar to [6] and addressing (**D2**) as well as (**D1**) and (**D4**). For spatial matches at the same time, a guide along the epipolar line is provided to the user. For temporal matches, no guides are possible since movement is not constrained in 3D space. The match does not need to be very precise, since it is used merely as initialization that is further refined by the data term. In practice, matches are often necessary for large displacements, and subsequently incur a strong smoothness penalty S_m, Eq. 5; this can be ameliorated by an additional edge scribble. It is also a good strategy to define matches on a coarse pyramid level L as early as possible to allow the data term to refine the match on finer levels.

For integration into the scene flow, each pixel $\mathbf{p}_s$ inside the source region is first related to a target pixel $\mathbf{p}_t$, Fig. 6, using an affine transform based on the user-defined translation, rotation and scale. In the case of motion, we project both source and target pixel into world space using the current z solution, yielding world space points $\mathbf{P}_s$ and $\mathbf{P}_t$, Fig. 6 (a); the difference between them is the new motion

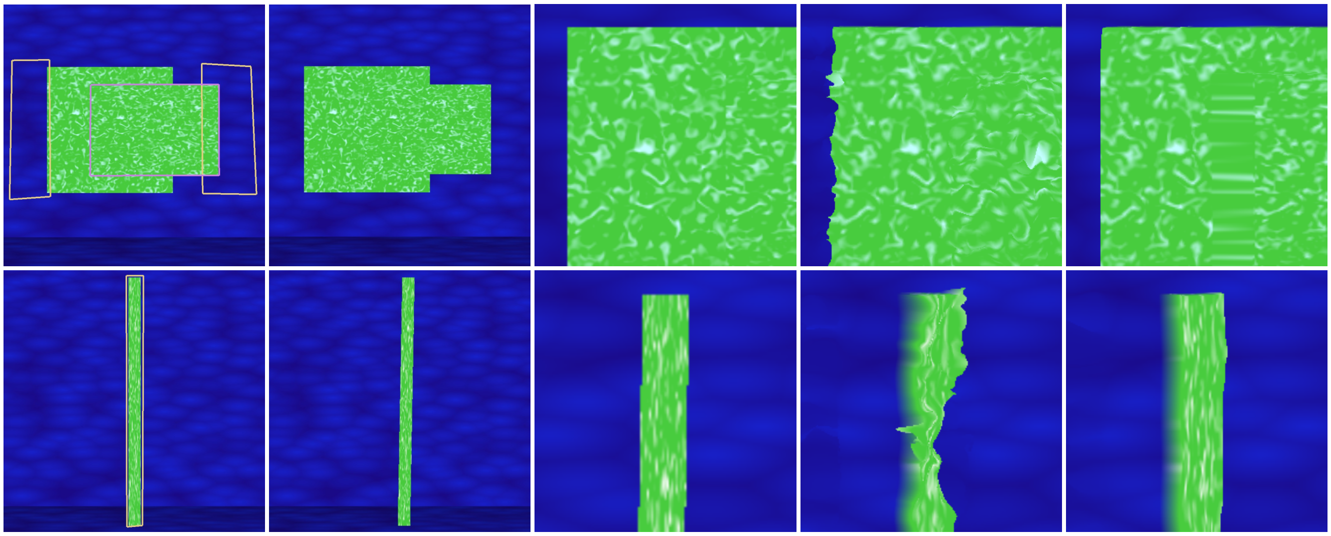

(a) user interaction at t_0 (b) reference image at t_1 (c) zoomed at t_1 (d) without user interaction (e) with user interaction

Figure 7: Warped images. Row 1: Correcting an ambiguous discontinuity (S1). As in Fig. 5 row 1, occluded regions are marked, and the edge allows formation of the discontinuity (a). Warped images should look like the reference at t_1 (b, c). Unaided scene flow results in craggedness around the discontinuity (d), while the corrected discontinuity is very even (e). Row 2: Preserving fine structures (S2). The stick receives an increased smoothness weight (a). Compared to the reference (b, c), automated scene flow often smooths the background flow over the stick (d). The corrected version preserves the stick structure (e).

vector $\mathbf{v}$. In the 1D constrained case of depth, we project camera rays $\mathbf{R}_s$ and $\mathbf{R}_t$ into world space, Fig. 6 (b), calculate the shortest line between the rays via closed form solution, and take the z component in the middle of the line.

At each outer fixed-point iteration, we locally set the current z and/or $\mathbf{v}$ solution to the calculated value. During matching, we repeat this up to a level L_d chosen by the user; this leads to a stable and predictable influence on the surrounding region similar to the strategy in [11] (however, we do not use an anisotropic term; note how the user displacements in [11] are only shown for regions with good object/background image gradients). The lower (finer) L_d, the less iterations will be performed to refine the match. Depending on the situation, this allows the user to specify either refined approximate or enforced precise matches.

3.5 Direct Matching

In addition to the intra-optimization tools above, we also allow the user to match regions after the estimation has finished, effectively emulating [6] for scene flows. However this is only needed for rare intractable cases (e.g. two grids moving against each other), requires very precise user input and was not used for our results. It also requires waiting for the final scene flow, whereas we usually run the last refinement levels unattended.

3.6 View Propagation Tools

When scene flows for multiple hero cameras are desired (e.g. to produce the morphed frames in our results), the first solution is to reproject our world space scene flow into the other cameras. However, pixels disoccluded in the target camera will be undefined, necessitating a new scene flow calculuation.

We therefore implemented the propagation of scribbles towards other cameras and time steps according to the estimated scene flow. Smoothness and matching regions are reprojected using z and $\mathbf{v}$ obtained at the centroid of the region. Edges must either partially enclose some region or be defined in pairs that, when connected, form a closed region whose centroid can be taken. Occlusion regions are often on the background, near a discontinuity; they must either be redefined or can be transferred by taking the centroid of a nearby smoothing or matching scribble.

In all cases, we allow further translation/rotation via mouse clicks and numpad keys. For spatially propagated scribbles, usually only a minority must be corrected and even fewer redefined. Temporally propagated scribbles suffer from missing temporal symmetry more often and thus need redefinition more frequently. Compared to manual redefinition of all scribbles, the propagation approach saved considerable time in the creation of our results.

4. RESULTS

Our results are presented in two parts, and shown in motion in our video[2]. Sec. 4.1 shows the hero camera frame at t_0 *warped* towards t_1. This approach is best suited for quality assessment because artifacts in the scene flow are directly identifiable. Sec. 4.2 presents results *morphed* from all cameras at both t_0 and t_1 towards a virtual camera position k_{vrt} and a time t_{vrt}, i.e. all images are warped towards the same virtual spatiotemporal location and then blended. This method produces the visually most pleasing results. We conclude with a quantitative evaluation of the rendered output and a summative user study.

4.1 Warps for Quality Assessment

For a visual assessment of flow field quality, we present the hero camera's t_0 frame warped fully towards t_1. By comparing the warped frame to the reference frame at t_1, flow field artifacts become directly visible. For warping, we use a fully connected mesh with one vertex in the center of each pixel; we also use nearest neighbor interpolation because single pixels can be discerned that way. To clearly demonstrate cause and effect, we present 6 synthetic examples with minimal user interaction, each addressing one of the failure cases identified in Sec. 2. Often, e.g., an additional

[2] Video and further paper details available under: `http://graphics.tu-bs.de/publications/ruhl2015acmmm/`

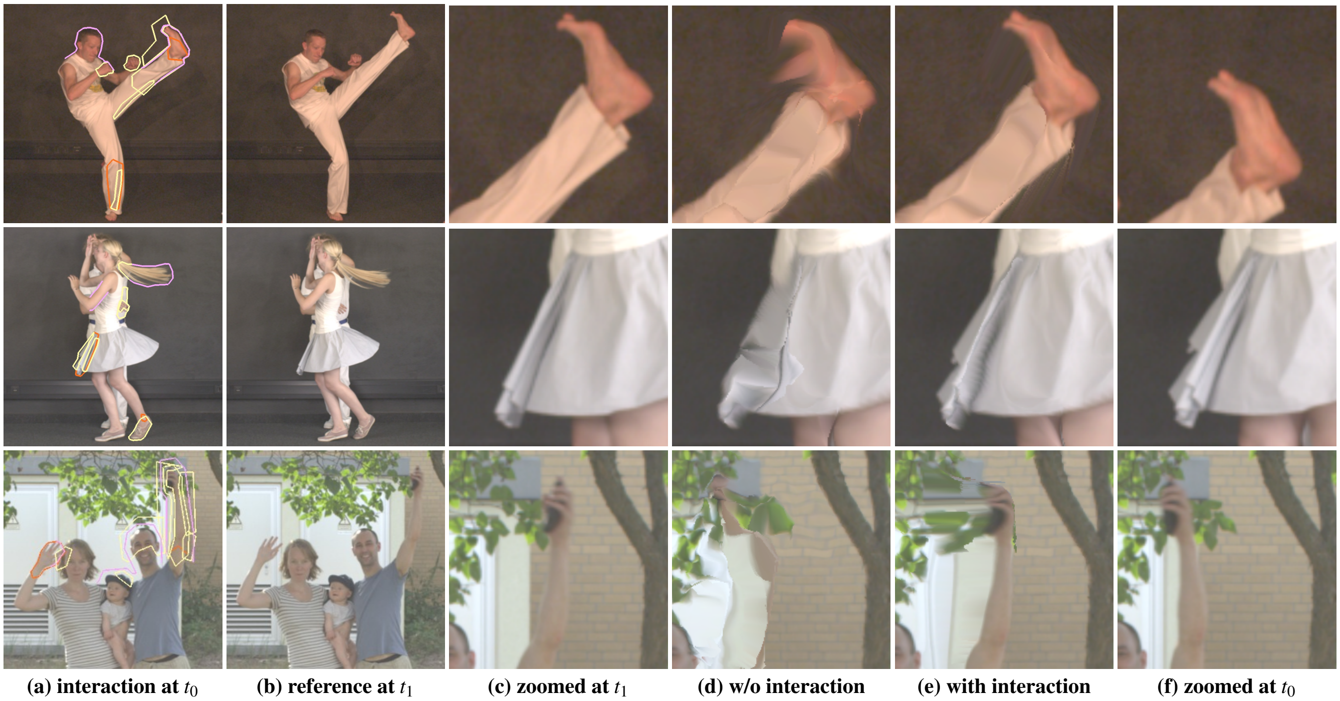


Figure 8: Warped images. Row 1: Capoeira scene with large displacements, low texture and shadows. The foot is matched, shadows and creases marked as occluded, smoothness increased (a). Compared to the reference at t_1 (b, c), the foot motion leads to streaking artifacts (d). The corrected version forms the foot better (e) given the large displacement from (f). Row 2: Dancing scene with creases in low textured clothing. A large crease is matched, the outer side marked occluded, smoothness increased (a). Compared to the reference (b, c), the skirt is not solved well at all (d). The corrected version resolves the motion plausibly (e), with some of the left border from t_0 (f) remaining. Row 3: Family scene with fine structures under complex background. The arm is matched, smoothed, and marked as occluded where color constancy violations remain (a). Compared to the reference (b, c), the arm is completely destroyed in (d). The corrected version preserves the shape of the arm (e) and stretches the disocclusion stemming from (f) evenly.

edge scribble could improve the result further but would reduce clarity. Additionally, we present 3 real-world examples, where all tools have been applied in combination.

Synthetic Examples. Fig. 4 row 1 shows a rotating textured sphere with a large specularity, addressing (**D1**). The color constancy violation is treated by specifying a full spatiotemporal occlusion (**OT**) that deactivates all brightness constancy terms $BC_{m,s1,s2}$, Eq. 2–4. Since the scene flow model does not separate light transport and object surface, moving the specularity with the object is the geometrically correct choice and the desired effect; alternatively, preserving the specular location would be possible with an edge scribble (**ET**) to separate it from the surrounding motion. The corrected sphere shows intact letters where the automated version shows distorted ones.

Fig. 4 row 2 shows three diffuse balls with the same texture, all moving to new locations, addressing (**D2**). The ambiguous matching problem is mitigated by providing three manual matches (**MT**), which are then refined automatically. Note that as no further edge scribbles have been applied to the example, the background is partly drawn with the balls. The corrected balls move to the correct location where the automated version shows severe misplacement.

Fig. 5 row 1 shows two textured boxes horizontally moving apart from each other, leading to occlusions of the background and a disocclusion of the red box, addressing (**D3**). The occlusion problem is solved by applying the temporal occlusion tool (**OT**) on the background. The scribbles may well encompass part of the foreground, as long as the marked foreground is spatially visible in other cameras. An additional edge scribble (**ET**) around the green box targets the disocclusion problem, and leads to a consistent discontinuity formation at the purple line. Note that had we relied on anisotropic smoothness, it would have lead to wavy edges as in (d) due to irregular brightness coincidences e.g. between box and background. The corrected boxes show straight borders and discontinuities where the automated version shows distorted ones.

Fig. 5 row 2 shows the same two moving boxes from row 1 but this time untextured, addressing (**D4**). While it is impossible to determine the scene flow quality visually in the middle of the image, the same two temporal occlusion scribbles (**OT**) and one edge scribble (**ET**) at some plausible box boundary solve the problem. Our tests confirm that the edge scribble is indeed necessary; without it, irregular smoothness propagation from the top and bottom of the boxes arrives as incoherent horizontal u motion at the left and right rectangle borders. The corrected box preserves the straight border where the automated version produces irregular curvature.

Fig. 7 row 1 is almost identical to Fig. 5 row 1 but this time the two boxes have the same texture and are much more difficult to disambiguate, addressing (**S1**). The same user interaction from Fig. 5 solves the issue here as well, since it allows flow field divergence at the correct location (**ET**) and disallows impossible pixel matches in the regions occluded at t_1 (**OT**). The corrected boxes preserve straight borders and discontinuity where the automated version shows distorted results.

Fig. 7 row 2 shows a thin, slowly moving structure that is prone to being overridden by the surrounding background due to its rela-

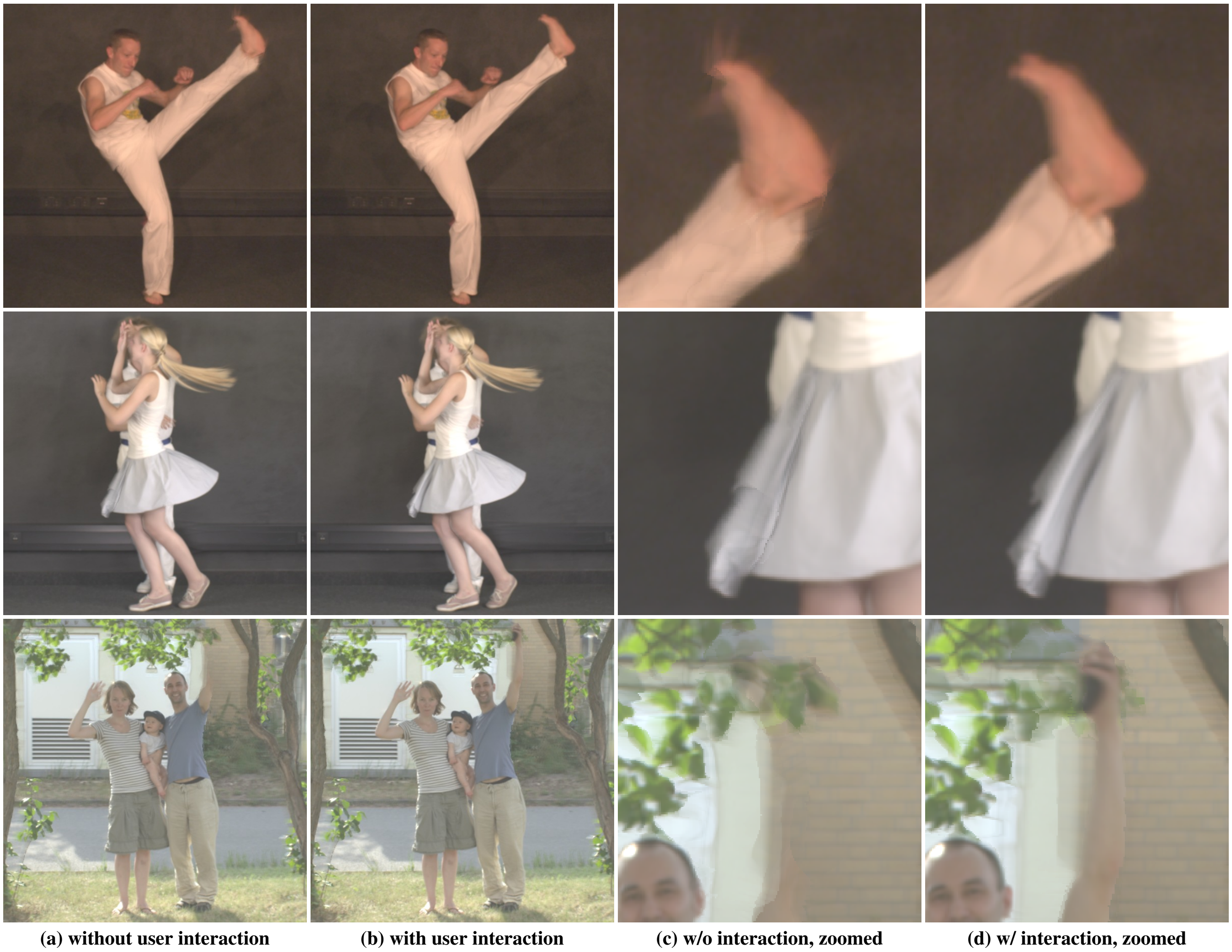

(a) without user interaction
(b) with user interaction
(c) w/o interaction, zoomed
(d) w/ interaction, zoomed

Figure 9: Morphed images from 8 views (4 cameras, 2 time steps). Row 1: Capoeira scene. Different artifacts in the views produce a halo streaking effect around the foot (c), while the corrected version features only minor motion blur (d). Row 2: Dancers scene. The blended skirt artifact (c) has vanished in the repaired version (d). Row 3: Family scene. Widely differing failure modes make the blended arm almost invisible (c), while the correct version preserves the arm's shape.

tively small influence in the smoothness term evaluation, addressing (**S2**). The problem is solved by demanding a large smoothness weight for the stick, using a region scribble around the structure (**ST**). Note that an additional edge scribble would reduce the impact of brightness coincidences between stick and background texture further. The corrected stick retains its shape while the automated version deforms its shape considerably.

Real-World Examples. We use all four tools in combination on our real-world footage, which was recorded with 4 RED Scarlet-X at 4K resolution and 15cm interocular baseline, only approximately color-graded, and downsampled to 540p to reduce noise.

Fig. 8 row 1 shows a Capoeira scene with fast motion and low-textured clothing featuring crease deformations and shadows. Consider the high-kicking leg. Foot and lower leg receive increased smoothness (**ST**) and an edge (**ET**) to allow large motions against a static background. The background above the leg is marked as temporally occluded (**OT**). An additional match (**MT**) around the ankle is required to overcome an incorrect local minimum. A clothing crease on the thigh not visible at t_1 is also marked as occluded (**OT**). Further edits include the standing leg with match and smoothing, the dark hair being edge-protected from the equally dark background, and hands being smoothed. The corrected foot shows a consistent shape where the automated version shows severe streaking artifacts.

Fig. 8 row 2 shows a pair-dance scene with actress/actor occlusions under same-colored clothing, a classic failure case for anisotropic smoothness, and deep clothing creases. Consider the left side of the skirt. The shadowed and deforming crease is matched (**MT**) and left and right side marked as occluded (**OT**) for true temporal occlusion and color constancy violation respectively. Further edits include smoothing the flowing hair and edge-protecting it against the background; edges around the female dancer's upper arm; increased smoothness and temporal occlusion at the male dancer's hand; and smoothing and matching around the foot and ankle. The corrected skirt shows a plausible shape where the automated version shows unrealistic folding.

Fig. 8 row 3 shows a complex outdoor family scene with sharp depth discontinuities as well as fine structures with large motions, which are usually lost in the downsampling of the image pyramid. Consider the rightmost arm, its color similarity to the right

Dataset	auto	user	RRE	Fig. 8
Capoeira (foot)	0.937	0.944	+12%	row 1 (c)–(e)
Dancers (skirt)	0.919	0.940	+26%	row 2 (c)–(e)
Family (right arm)	0.879	0.880	+1%	row 3 (c)–(e)

Table 1: Structural similarity index [19] between the reference frame at t_1 and the images warped to t_1 either without (auto) or with (user) correction. While both scores are already very good, user interaction yields a further reduction of the remaining error (RRE) of up to 26%.

background, and the hand's large displacement relative to its size. The arm is matched (**MT**), smoothness increased (**ST**), and edge-protected (**ET**). Due to the texture similarity of arm and background, data term refinement after matching can still produce artifacts, which are suppressed with the occlusion tool (**OT**). Note that the smearing artifacts in the disoccluded region on the left side of the arm are caused by the fully connected mesh used in our warping approach. Further edits include smoothness and edge-protection around the heads; and match, small edge and data term deactivation via occlusion to repair the leftmost hand. The corrected arm preserves its shape where the uncorrected version tears the arm apart.

In all examples, editing times are on the order of minutes; applying the scribbles is a matter of seconds, and observing the effect forming in the ongoing optimization takes tens of seconds over several re-warp iterations. While we already run a GPU scene flow to achieve interactive feedback, an even faster scene flow algorithm or faster GPUs would reduce total editing time further.

4.2 Morphs for Rendering

In the following, we present how our tools can be used to improve image-based rendering quality, morphing 8 views (4 cameras at 2 time steps) based on Lumigraph rendering [2]. The virtual camera is defined by a virtual camera position $k_{\mathrm{vrt}} = 0..3$ and virtual time $t_{\mathrm{vrt}} = 0..1$. Frames are blended with linear temporal weighting and spatial weighting depending on the viewing angle per pixel (details in [2]). The best virtual spatiotemporal position to observe artifacts is in the middle between two cameras and times, i.e., $t_{\mathrm{vrt}} = 0.5$ and $k_{\mathrm{vrt}} = 0.5$, 1.5 or 2.5 respectively, where input from the 4 adjacent views are maximally warped before being blended. Below, we show results at $t_{\mathrm{vrt}} = 0.5$ and $k_{\mathrm{vrt}} = 0.5$.

Fig. 9 row 1 shows the improved visual quality of morphs with user interaction compared to morphs without user interaction, mirroring the improvements of warped results, c.f. Fig. 8. On the Capoeirista's high foot, the 4 involved views all have differing artifacts, each of which are 25% visible as a halo artifact when blended. The corrected version is spatiotemporally consistent and therefore able to provide high-quality blending.

Fig. 9 row 2 again shows that user interaction, here on the skirt, solves the shortcomings of automated estimation. The left side of the Dancer's skirt blends 4 different artifacts in the unaided case, which are replaced with a consistent appearance in the corrected version.

Fig. 9 row 3 shows the most extreme example. Due to widely differing arm motions in the 4 automated estimates, the arm effectively vanishes during blending. In contrast, the corrected version leaves the arm entirely intact. Note that the smearing artifacts in the disoccluded region left of the arm are rendering artifacts due to warping with a fully connected mesh. In future work, a more sophisticated rendering removing these regions in the relevant views will improve morphing results.

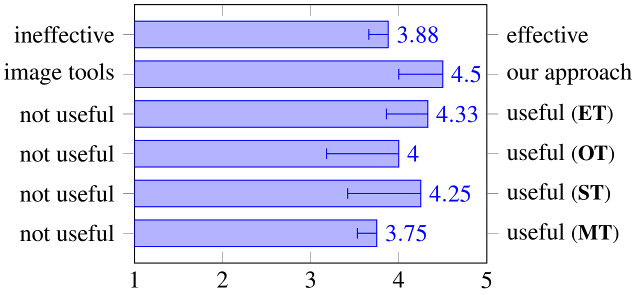


Figure 10: User study with scores $1..5$**; mean values and standard deviation are shown. Sample size was 4 experts with both image editing and stereo and/or optical flow experience. Questions were general usefulness, relative usefulness of our tool compared to image-based tools, and usefulness per tool.**

4.3 Quantitative Evaluation

In addition to the visual quality, we assess the numerical quality of our corrected scene flows compared to uncorrected ones based on the structural similarity index method (SSIM) [19], comparing fully-warped images to the reference image at t_1, Table 1. We report scores calculated on the partial images shown in Fig. 8 (c)–(e).

User interaction for the Capoeira and Dancers scenes removes 12–26% of the remaining error. In contrast, the Family scene shows only a 1% reduction, though the right-arm artifact is the visually most obvious and disturbing of all examples. The close scoring is probably caused by the color similarity of arm and background.

4.4 Summative Evaluation

Since there are no multimedia authoring tools using interactive scene flow estimation, we evaluate the relative attractiveness of our approach against the industry standard, fixing rendered frames with image-space tools. Our tools are meant for trained visual artists and not for the general public. As such, we undertook a summative evaluation with 4 experts in the age range 25–35 with at least 5 years of image processing experience as well as exposure to stereo and/or optical flow. They were coached in the use of our tools for up to 20 minutes each, on training footage not used for subsequent evaluation, the latter of which used the Capoeira scene, Fig. 8–9 row 1. Assuming a moderate number of 10 frames rendered from an automatically estimated scene flow at $t_{0.0}$, $t_{0.1}$, $t_{0.2}$, .., $t_{1.0}$, we asked our experts to repair the scene flow for subsequent re-rendering. For the alternative workflow, we asked them to repair the 10 originally rendered frames instead, using an image-space tool of their choice (all chose Adobe Photoshop). At 10 minutes into either task, the experts were instructed to finish their last operation. Afterwards, the results of both workflows were compared and the experts were asked to rate both with 1 (not useful) to 5 (very useful) on the MOS (mean opinion score) scale.

As shown in Fig. 10, the general usefulness of our approach for the given task was confirmed with a mean score of 3.88. All experts expressed the wish for real-time scene flow recomputation after each scribble, which is not yet possible with state-of-the-art algorithms. Single wishes included using scribbles on target frames instead of the source frame, and instant comparison against the effect of previous scribbles. Compared to image-space tools, the relative attractiveness of our method increased to a mean score of 4.50, with all experts seeing the built-in spatiotemporal consistency between frames as a key advantage of our method.

The experts found all four tools similarly useful with mean scores ranging from 3.75–4.33; all noted that 20 minutes of coaching were sufficient to use the tools effectively. Given the fact that they are used to Photoshop, all experts noted that an increasing familiarity with our tools would probably allow even better results.

5. DISCUSSION

All experts agreed on the effectiveness of our approach as demonstrated by the improved visual quality of the output frames. The most desired improvement were instant response times, which requires realtime scene flow algorithms. With respect to the latter, our scene flow optimization could potentially benefit from a primal-dual approach in the style of [22], left for future work.

Our scene flow estimation algorithm is a GPU-based re-implementation of [1]. By integrating user guidance, we expect similar gains for other scene flow approaches since the failure modes are based on commmon assumptions and scene properties rather than algorithmic intricacies.

When used for visual media production, our approach does not preclude the use of image-based tools; it optionally preceeds it, reducing but in some cases not fully mitigating image-space work.

Our content production approach has two time savers: First the approximate way most scribbles can be defined (edges between two salient regions being the sole exception), and second the arbitrary number of frames that can be rendered using a single scene flow field. Additionally, when going from traditional multimedia production towards free spacetime navigation, "post-production frame correction" is not possible because the number of frames is arbitrary. In this case, editing depth and motion itself is the only viable way to produce artifact-free output frames.

6. CONCLUSIONS

In this paper we presented a workflow for high-quality view interpolation with interactive scene flow estimation, useful for spatiotemporal multimedia content authoring. We analyzed common artifacts during scene flow estimation and presented four interactive tools to overcome these. By refining the coarse user input to subpixel precision using a variational formulation, we ease the user of the burden of precise and time-consuming interaction. This vastly decreases the required effort compared to the common technique of repairing each rendered output frame by hand. Output rendered with our corrected scene flow appears plausible to a human observer, making our approach suitable for applications such as virtual spacetime navigation or high-quality image-based rendering.

7. ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union's Seventh Framework Programme FP7/2007-2013 under grant agreement no. 256941, Reality CG. We would also like to thank Basha et al. [1] for making the multi-view scene flow C++ source code and part of their data sets openly available.

8. REFERENCES

[1] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *International journal of computer vision*, 101(1):6–21, 2013.

[2] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph rendering. In *Proc. of 28th SIGGRAPH*, pages 425–432. ACM, 2001.

[3] G. Chaurasia, O. Sorkine, and G. Drettakis. Silhouette-aware warping for image-based rendering. In *22nd Eurographics conference on Rendering*, pages 1223–1232, 2011.

[4] Y. Doron, N. D. Campbell, J. Starck, and J. Kautz. User directed multi-view-stereo. In *2nd Workshop on User-Centred Computer Vision (at ACCV)*, 2014.

[5] M. Guttmann, L. Wolf, and D. Cohen-Or. Semi-automatic stereo extraction from video footage. In *Intl. Conference on Computer Vision (ICCV)*, pages 136–142, 2009.

[6] F. Klose, K. Ruhl, C. Lipski, and M. Magnor. Flowlab - an interactive tool for editing dense image correspondences. In *European Conf. on Visual Media Production (CVMP)*, 2011.

[7] C. Linz, C. Lipski, L. Rogge, C. Theobalt, and M. Magnor. Space-time visual effects as a post-production process. In *ACM Multimedia 2010 - 1st Intl. 3DVP Workshop*, 2010.

[8] P. Mordohai. On the evaluation of scene flow estimation. In *Computer Vision – ECCV 2012. Workshops and Demonstrations*, pages 148–157. Springer, 2012.

[9] D. Ring and A. Kokaram. User-assisted feature correspondence matching. In *Proc. European Conference on Visual Media Production*, pages 214–219, 2009.

[10] K. Ruhl, M. Eisemann, and M. Magnor. Cost volume-based interactive depth editing in stereo post-processing. In *Euro. Conference on Visual Media Production (CVMP)*, 2013.

[11] K. Ruhl, B. Hell, F. Klose, C. Lipski, S. Petersen, and M. Magnor. Improving dense image correspondence estimation with interactive user guidance. In *Proc. ACM Multimedia 2012*, pages 1129–1132. ACM, Oct. 2012.

[12] J. Steurer. Tri-focal rig (practical camera configurations for image and depth acquisition). *SMPTE Conferences*, 2013(10):1–15, 2013.

[13] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2439. IEEE, 2010.

[14] D. Sykora, D. Sedlacek, S. Jinchao, J. Dingliana, and S. Collins. Adding depth to cartoons using sparse depth (in)equalities. *Computer Graphics Forum*, 29(2), 2010.

[15] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *7th Intl. Conf. Computer Vision (ICCV)*, volume 2, pages 722–729. IEEE, 1999.

[16] C. Vogel, K. Schindler, and S. Roth. Piecewise rigid scene flow. In *International Conference on Computer Vision (ICCV)*, pages 1377–1384. IEEE, 2013.

[17] L. Wang and R. Yang. Global stereo matching leveraged by sparse ground control points. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3033–3040, 2011.

[18] O. Wang, M. Lang, M. Frei, A. Hornung, A. Smolic, and M. Gross. Stereobrush: interactive 2d to 3d conversion using discontinuous warps. In *8th Eurographics Symposium on Sketch-Based Interfaces and Modeling*, pages 47–54, 2011.

[19] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4), 2004.

[20] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 optical flow. In *British Machine Vision Conference (BMVC)*, volume 34, 2009.

[21] L. Wilkes. The role of Ocula in stereo post production. *The Foundry, Whitepaper*, 2009.

[22] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Pattern recognition: 29th DAGM symposium*, volume 29, pages 214–223, 2007.

[23] C. Zhang, B. Price, S. Cohen, and R. Yang. High-quality stereo video matching via user interaction and space-time propagation. In *Intl. Conf. 3DV*, pages 71–78. IEEE, 2013.