Computational Visual Media
https://doi.org/10.1007/s41095-017-0089-1 Vol. 4, No. 1, March 2018, 3–15

Research Article

Surface tracking assessment and interaction in texture space

Johannes Furch[1], **Anna Hilsmann**[1,2], **and Peter Eisert**[1,2] (✉)

Abstract In this paper, we present a novel approach for assessing and interacting with surface tracking algorithms targeting video manipulation in post-production. As tracking inaccuracies are unavoidable, we enable the user to provide small hints to the algorithms instead of correcting erroneous results afterwards. Based on 2D mesh warp-based optical flow estimation, we visualize results and provide tools for user feedback in a consistent reference system, texture space. In this space, accurate tracking results are reflected by static appearance, and errors can easily be spotted as apparent change. A variety of established tools can be utilized to visualize and assess the change between frames. User interaction to improve tracking results becomes more intuitive in texture space, as it can focus on a small region rather than a moving object. We show how established tools can be implemented for interaction in texture space to provide a more intuitive interface allowing more effective and accurate user feedback.

Keywords surface tracking; assessment; interaction; mesh warp; optical flow

1 Introduction

Elaborate video manipulation in post-production often requires means to move image overlays or corrections along with the apparent motion of an image sequence, a task termed *match moving* in the visual effects community. The most common commercial tracking tools available to extract the apparent motion include manual keyframe warpers, point trackers, dense vector generators (for optical flow), planar trackers, keypoint match-based camera solvers (for rigid motion), and 3D model-based trackers [1–3]. Many of these tools allow for some kind of user interaction to guide, assist, or improve automatically generated results. However, while increasingly being discussed in the research community [4–6], visual effects artists have not yet adopted user interaction with dense optical flow-based estimation methods. We believe this is due to the technical aims of most proposed tools, their relative complexity of usage, and the difficulty of assessing tracking quality in established result visualizations.

In this paper, we introduce the concept of assessment and interaction in texture space for surface tracking applications. We believe the quality of a tracking result can best be assessed on footage mapped to a common reference space. In such a common space, perfect motion estimation is reflected by a perfectly static sequence, while any apparent motion suggests errors in the underlying tracking. Furthermore, this kind of representation allows for the design of tools that are much simpler to use, since even in case of errors, visually related content is usually mapped in close spatial proximity throughout the sequence. Interacting with the tracking algorithms directly and improving the tracking results instead of adjusting the overlay data have the clear advantage of decoupling technical aspects from artistic expression.

2 Related work and contribution

Today, many commercial tools exist for motion extraction. These tools often allow user interaction to guide, assist, or improve automatically generated

1 Fraunhofer HHI, Berlin, 10587, Germany. E-mail: peter.eisert@hhi.fraunhofer.de (✉).
2 Humboldt University, Berlin, 10099, Germany.
Manuscript received: 2017-03-06; accepted: 2017-04-29

results. However, many commercial implementations are limited to simple pre- and post-processing of input and output respectively [7, 8]. Also, often the motion estimation is based on key point-based trackers. These methods allow for the estimation of rigid, planar, or coarse deformable motion only, as they are based on sparse feature points and merely contribute a limited number of constraints to the optimization framework. In contrast, dense or optical flow-based methods use information of all pixels in a region of interest and therefore allow for much more complex motions. However, user interaction has not yet been integrated into dense optical flow-based estimation methods in commercial tools.

In the research community, a variety of user interaction tools for dense tracking and depth estimation have been proposed in recent years. One possibility is to manually correct the output of automatic processing and then to retrain the algorithm as is for example done in Ref. [9] for face tracking. In order to avoid tedious manual work when designing user interaction tools, one important aspect is to find a way to also integrate inaccurate user *hints* directly into the optimization framework that is used for motion or depth estimation. Inspired by scribble-based approaches for object segmentation [10], recent works on stereo depth estimation have combined intuitive user interaction with dense stereo reconstruction. While Zhang et al. [6] directly work on the maps by letting the user correct existing disparity maps on key frames, other approaches work in the image domain and use sparse scribbles on the 2D images to define depth layers, using them as soft constraints in a global optimization framework which propagates them into per-pixel depth maps through the whole image or video sequence [11, 12]. Similarly, other approaches use simple paint strokes to let the user set smoothness, discontinuity, and depth ordering constraints in a variational optimization framework [4, 5, 13].

In this work, we address user assisted deformable video tracking based on mesh-based warps in combination with an optical flow-based cost function. Mesh-based warps and dense intensity-based cost functions have already been applied to various image registration problems, e.g., in Refs. [14, 15], and have been extended by several authors to non-rigid surface tracking in monocular video sequences [16, 17]. These approaches can estimate complex motions and deformations but often fail in certain situations, like large scale motion, motion discontinuity, correspondence ambiguity, etc. Here, user hints can help to guide the optimization. In our approach, we integrate user interaction tools directly into an optimization framework, similarly to the approach in Ref. [16] which not only estimates geometric warps between images but also photometric ones in order to account for lighting changes. Our contribution lies on one hand in illustrating how texture space in combination with a variety of change inspection tools provides a much more natural visualization environment for tracking result assessment. On the other hand, we show how tools similar to those other authors have introduced can be redesigned and adapted to create powerful editing instruments to interact with the tracking results and algorithms directly in texture space. Finally, we introduce an implementation of our texture space assessment and interaction framework.

3 Surface tracking

3.1 Model

Given a sequence of images $I_0, \ldots, I_N$, without loss of generality we assume that I_0 is the reference frame in which a region of interest R_0 is defined. Furthermore, it is assumed that the image content inside this reference region represents a continuous surface. The objective is to extract the apparent motion of the content inside R_0 for each frame. We determine this motion by estimating a bijective warping function $W_{0i}(\boldsymbol{x}_0; \theta_{0i})$ that maps 2D image coordinates $\boldsymbol{x}_0 \in R_0$ to $\boldsymbol{x}_i$ in a region R_i in I_i based on a parameter vector θ_{0i} describing the warp. The inverse of this function $W_{0i}^{-1} = W_{i0}$ is defined for the mapped region R_i. As the indices of $\boldsymbol{x}$ and θ can be deduced from W, they will be omitted in the following.

We design the bijective warping function based on deforming 3D meshes $M(V, T)$. The meshes consist of a consistent triangle topology T and frame dependent vertex positions $\boldsymbol{v} \in V$. Coordinates are mapped from R_i to R_j based on barycentric interpolation of the offsets $\Delta\boldsymbol{v} = \boldsymbol{v}_i - \boldsymbol{v}_j$ between

the meshes M_i and M_j covering the regions:

$$W_{ij}(\boldsymbol{x};\theta) = \boldsymbol{x} - \sum_{l=1}^{3} \beta_l(\boldsymbol{x}) \Delta \boldsymbol{v}^l(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{B}(\boldsymbol{x})\theta \quad (1)$$

where $\boldsymbol{B}(\boldsymbol{x}) \in \mathbb{R}^{2\times 2|V|}$ is a matrix representation of the barycentric coordinates β, $\boldsymbol{v}^l$ are the vertices of the triangle containing $\boldsymbol{x}$, and $\theta \in \mathbb{R}^{2|V|\times 1}$ is the parameter vector containing all vertex offsets in x and y directions.

The mapping defined in Eq. (1) reflects the rendering of object mesh M_i into image I_i based on texture coordinates defined by the vertices of M_j for the object texture I_j, i.e., $I_i(\boldsymbol{x}) = I_j(W_{ij}(\boldsymbol{x};\theta))$. Therefore, the objective can be reformulated as the recovery of model parameters from a rendered sequence in which I_0 represents the texture and the vertices of M_0 represent the texture coordinates.

The sequence tracking problem can be interpreted as the requirement to find a set of mesh deformations $M_1, \ldots, M_N$ of a reference mesh M_0 that minimizes each difference $I_0(W_{i0}(\boldsymbol{x};\theta)) - I_i(\boldsymbol{x})$ for coordinates $\boldsymbol{x} \in R_i$. The free parameters in this equation are the vertex offsets that can be changed by adapting the positions of the meshes M_i. Note that the motion vectors for pixel positions in R_0 are implicitly estimated, since the inverse warping function W_{0i} can be constructed by swapping the two meshes.

A warping function that maps image I_j to image I_i can be found by minimizing the following objective:

$$E_{\mathrm{D}}(\theta) = \frac{1}{|R_i|} \sum_{\boldsymbol{x}\in R_i} \psi\left(I_j(W_{ij}(\boldsymbol{x};\theta)) - I_i(\boldsymbol{x})\right) \quad (2)$$

where ψ is a norm-like function (e.g., SSD, Huber, or Charbonnier). The pixel difference is normalized by the pixel count $|R_i|$, so the function cannot be minimized by shrinking the region. In addition, the function can be used across different scales of a Gaussian pyramid. Motion blur can also be explicitly considered by adding motion dependent blurring kernels to the data term [18].

To tackle noisy image data and to propagate motion information for textureless areas, we constrain the permitted deformation of the mesh by introducing a uniform mesh Laplacian $\boldsymbol{L}$ as a smoothing regularizer based on mesh topology, and include it into our objective as an additional term $E_{\mathrm{L}}(\theta)$. The final nonlinear optimization problem is as follows:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \left(\lambda_{\mathrm{D}} E_{\mathrm{D}}(\theta) + \lambda_{\mathrm{L}} E_{\mathrm{L}}(\theta)\right) \quad (3)$$

where λ_{L} balances the influence of the terms involved and is set to a multiple of $|V|/|R_j|$, so the influence of the Laplace term is scaled by the average amount of per triangle image data.

Using the Gauss–Newton algorithm, the parameter update $\theta_{k+1} = \theta_k + \Delta\theta$ to iteratively find $\hat{\theta}$ is determined by solving equations that require the Jacobian of the residual term. The Jacobian $J_{\mathrm{D}} \in \mathbb{R}^{|R|\times 2|V|}$ of the data term is

$$\frac{\delta E_{\mathrm{D}}(\theta_k)}{\delta\theta} = -\frac{B(\boldsymbol{x})}{|R_i|} \nabla I_j\left(W_{ij}(\boldsymbol{x};\theta_k)\right)$$

where $\nabla I \in \mathbb{R}^{|R|\times 2}$ is the spatial image gradient in x and y directions.

For a more detailed discussion of the theory behind image registration using mesh warps we refer to Ref. [19].

3.2 Photometric registration

The tracking method described above makes use of the brightness constancy assumption, explaining all changes between two images by the pure geometric warp W_{ij}^{g} in Eq. (1). Varying illumination and view-dependent surface reflection cannot be described by this model. In order to deal with such effects as well, we add a photometric warp:

$$W_i^{\mathrm{p}}(\boldsymbol{x};\theta^{\mathrm{p}}) = \sum_{l=1}^{3} \beta_l(\boldsymbol{x}) \rho^l(\boldsymbol{x}) = \boldsymbol{B}^{\mathrm{p}}(\boldsymbol{x})\theta^{\mathrm{p}} \quad (4)$$

that models spatially varying intensity scaling of the image. ρ^l is the scaling factor corresponding to vertex $\boldsymbol{v}^l$ which is related to the scaling of pixel $\boldsymbol{x}$ via the barycentric coordinates stored in $\boldsymbol{B}^{\mathrm{p}}$. This photometric warp represented by parameters θ^{p} is multiplicatively included in the data term in Eq. (2), leading to

$$E_{\mathrm{D}}(\theta) = \frac{1}{|R_i|} \sum_{\boldsymbol{x}\in R_i} \psi\left(W_i^{\mathrm{p}}(\boldsymbol{x};\theta^{\mathrm{p}}) \cdot I_j(W_{ij}^{\mathrm{g}}(\boldsymbol{x};\theta^{\mathrm{g}})) - I_i(\boldsymbol{x})\right)$$

This data term is solved jointly for the geometric and photometric parameters θ^{g}, θ^{p} in a Gauss–Newton framework [16]. Like for the geometric term, shading variations over the surface are constrained by a uniform Laplacian on the photometric warp.

3.3 Expected problems

To design meaningful interaction tools, it is necessary to understand what problems are to be expected by a purely automatic solution for determining the meshes $M_1, \ldots, M_N$. There are two distinct sources of error:

- The assumption that change can be modeled by geometric displacement (and smooth photometric adjustment) does not hold for most real-world scenarios. Since the appearance of the content in R_0 might vary significantly throughout the sequence (e.g., reflections, shadows, ...), the minimum of the objective function may not be close to zero.
- Every automated algorithmic solution has its own inherent problems. In our case, the optimization is sensitive to the initialization of the meshes M_i and while being easy to implement, a global Laplacian term that assumes constant smoothness inside the region of interest cannot model complex motion properties of a surface.

We use a number of heuristics to address these anticipated problems. First and foremost, we make use of the a priori knowledge that visual and therefore geometric change between adjacent frames is small. Therefore, starting at M_1, we iteratively determine M_i in the sequence using M_{i-1} as initialization for the optimization. Furthermore, assuming that M_{i-1} describes an almost perfect warping function to the reference frame, we use I_{i-1} (and therefore M_{i-1}) rather than I_0 as an initial image reference for optimizing M_i. However, to avoid error propagation (i.e., drift), we optimize with reference to I_0 (and therefore M_0) in a second pass using the result of the first pass as initialization. To deal with large frame to frame offsets, we run the optimization on a Gaussian image pyramid starting at low resolution. This problem can also be addressed by incorporating keypoint or region correspondences into the initialization or the optimization term [20, 21], an approach we adopt in a variety of ways for user interaction below. We address the problem of noisy data and model deviations by applying robust norms in E_{D} and E_{L}. Those problems have also been addressed by other authors by introducing a data-based adaption of the smoothness term to rigid motion [22]. In some cases violations of the brightness constancy constraint can be effectively handled by introducing gradient constancy into E_{D} [23].

4 Assessment and interaction tools

While the above optimization scheme generally yields satisfactory results, sometimes the global adjustment of parameters leaves tracking errors in a subset of frames. As our framework iteratively determines meshes M_i, it allows online assessment of the results. Therefore, whenever a problem is apparent to the user, the user can stop the process and interact directly with the algorithms using the tools described below. The optimization for a frame can be iteratively rerun based on additional input until a desired solution is reached. Therefore, the user can also decide what level of quality is needed and only initiate interaction if the currently determined solution is insufficient. Although each mesh M_i is ultimately registered to the reference image, reoptimization based on user input can lead to sudden jumps in the tracking. Such interruptions can easily be detected in texture space, and can usually be dealt with by back propagating the improved result and reoptimizing.

To be able to make use of established post-production tools, we have implemented our tracking framework as a plugin for the industry standard compositing software NUKE [2]. For illustrations in this section we use the public *Face Capture* dataset [24], while additional results on other sequences are presented in Section 5 and the accompanying video in the Electronic Supplementary Material (ESM). Assessment is best done by playing back the sequences.

4.1 Parameter adjustment

A number of concepts we introduced in the previous section can be fine-tuned by the user by adjusting a number of settings. While some parameters need to be fixed before tracking starts (e.g., the topology of the 2D mesh), most of them can be individually adjusted per frame. This includes the choice of the norm-like function, the λ parameters of the objective function in Eq. (3), the scales of the image pyramid, the images used as reference, and the mesh data to be propagated. This per-frame application implies that readjustment of a single frame with different parameter settings is possible, making the parameter adjustment truly interactive. In this context, the data propagation mode is an essential parameter: while the default mode is to propagate tracking data from the previous frame (i.e., to use M_{i-1} as initialization for M_i), if results from previous iterations are to be refined, M_i itself is used as

initialization. Given the implementation in a post-production framework, keyframe animation of the parameters using a number of interpolation schemes and linking them to other parameters are useful mechanisms. A possible application of this feature would be to link the motion of a known camera to the bottom scale of the image pyramid.

4.2 Texture space assessment

We call the deformation of image content in R_i to the corresponding position in R_0 the *texture unwrap* of R_i. Consequently, we say that the image information deformed in this way is represented in *texture space* and that an *unwrapped sequence* consists of a texture unwrap of all frames in the sequence (see rows 2–4 of Fig. 1). This terminology is derived from the assumption that the input sequence can be seen as a rendering of textured objects and that the reference frame provides a direct view onto the object of interest, so that image coordinates are interpreted as the coordinates of the texture. While the reference frame is usually chosen to provide good visualization, any mapping of those coordinates can also be used as texture space. Conversely, we say that image information (e.g., an overlay) that is mapped from R_0 to R_i is *match moved* (see row 5 in Fig. 1).

Traditionally, results are evaluated by watching a composited sequence incorporating match moved overlays, like the content of the reference frame, a checkerboard, or even the final overlay. In a way, this approach makes sense, since the result is judged by applying it to its ultimate purpose. However, since it is hard to visually separate underlying scene motion from the tracking, it is hard for a user to localize, quantify, and correct an error even if it can be seen that "something is off". So while viewing the final composite is a good way to judge whether the tracking quality is sufficient, it is not a good reference to assess or improve the quantitative tracking result: if presented with the match-moved content in row 5 of Fig. 1 in a playback of the whole sequence, an untrained observer would find it hard

Fig. 1 Visualizations of tracking results. The first row: samples from the public *Face Capture* sequence [24]. Rows 2–4: the unwrapped texture with and without shading compensation, and composited onto the reference frame. Bottom row: a match-moved semi-transparent checkerboard overlay.

to point out possible errors. Note that the content of the reference region is moving and deforming considerably, making the chosen framing the smallest possible to include all motion.

The main benefit of assessment in texture space is the static appearance of correct results. When playing back an unwrapped sequence, the user can zoom in and focus on a region of interest in texture space, and does not have to follow the underlying motion of the object in the scene. In this way, any change can easily be localized and quantified even by an untrained observer. Figure 1 illustrates in rows 2–4 different visualizations of the unwrapping space. The influence of photometric adjustment (estimated as part of our optimization) becomes very clear when comparing rows 2 and 3. Row 4 shows how layering the unwrapped texture atop the reference frame can help to detect continuity issues in regions bordering the reference region (e.g., on the right side of frame 200).

While a side-by-side comparison is not particularly well suited for assessment, errors are highlighted very clearly in Fig. 2. The depicted visualizations facilitate a variety of tools available in established post-production software for assessing change between images, mainly designed for color grading, sequence alignment, and stereo film production. The first three columns show comparisons between the reference image and the texture unwrap of the current image. For the shifted difference, we used the shading compensated unwrap to better highlight the geometric tracking issues. This illustration shows the difference between the two images with a median grey offset, highlighting both negative and positive outliers. This is particularly useful, as these positive and negative regions must be aligned

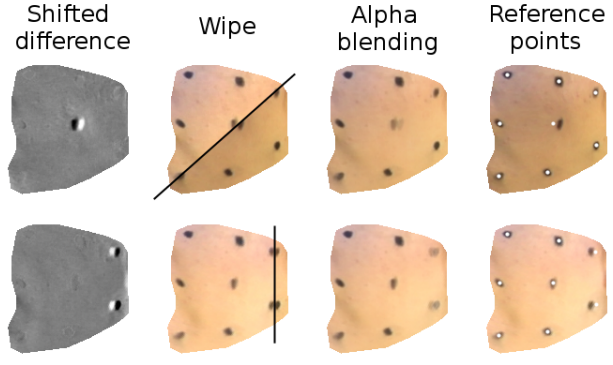

Fig. 2 Assessment tools. Top: frame 156, bottom: frame 200, in Fig. 1.

to yield the correct tracking result. Being part of our objective function, image differences are a perfect way of visualizing change. Furthermore, basic image analysis instruments like histograms and waveform diagrams can provide useful additional visualization to detect deviations in a difference image. A wipe allows the user to cut between the images at arbitrary positions, showing jumps if they are not perfectly aligned. Blending the same two images should result in an exact copy of the input. Therefore, if the blending factor is modulated, a semi transparent warping effect indicating the apparent motion between the two images can be observed. The last column in Fig. 2 illustrates a reference point assessment tool implemented as part of the correspondence tool introduced below. The user can specify the position of a distinct point $\boldsymbol{x}_{\text{ref}}$ in the reference frame, which is then marked by a white point. As the apparent position of any texture unwrapping of the corresponding image data should fall in the exact same location, visualizing this position as a point overlay throughout the sequence is very helpful for detecting deviations. It can also be used in combination with any of the other assessment tools. If a user detects a deviation, any available tool below can be applied to correct the error by aligning the content with the overlaying point without the need to revisit the actual reference image data.

In the following discussion of interaction tools, it is required in some cases to transform directional vectors from coordinates in texture space to those in the current frame. As the warping function is a nonlinear mapping, this transformation is achieved by mapping the endpoints of the directional vector:

$$\Delta \boldsymbol{x}_{\text{src}} = W_{0i}(\boldsymbol{x}_{\text{dst}}) - W_{0i}(\boldsymbol{x}_{\text{src}}) \tag{5}$$

4.3 Adjustment tool

The adjustment tool is an interactive user interface to correct an erroneous tracking result M_i for a single frame (see Fig. 3). The tool produces results in real time and any of the assessment tools introduced

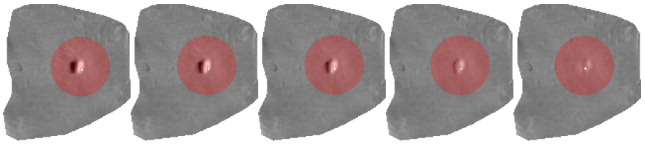

Fig. 3 Adjustment tool. The user drags the content to the correct location in texture space. For each mouse move event, real-time optimization is triggered and the result is updated. The radius of influence (i.e., the affected region) is marked in red.

above can be used for visualization. To initiate a correction, the user clicks on misplaced image content $\boldsymbol{x}_{\text{start}}$ in the unwrapped texture and drags it to the correct position $\boldsymbol{x}_{\text{end}}$ in the reference frame. Note that both of these coordinates are defined in texture space. Using the mouse wheel, the user can define an influence radius r visualized by a translucent circle around the cursor to determine the area that is influenced by the local adjustment. Whenever a mouse move or release event is triggered, the current position is set to be $\boldsymbol{x}_{\text{end}}$ and the mesh and therefore the assessment visualization is updated, so the user can observe the correction in real time. This interactive method is well suited to correcting large scale deviations from the desired tracking result, e.g., if the optimization is stuck in a local minimum. However, as it does not incorporate the image data, fine details are best left to the data-based optimization. So, while this corrected result could be kept as it is, it makes sense to use it as initialization for another data-based optimization pass.

The algorithmic correction of the mesh coordinates M_i of the current frame, the points $\boldsymbol{x}_{\text{start}}$ and $\boldsymbol{x}_{\text{end}}$ are transformed for processing using the warping function W_{0i} that is based on M_i at the time the correction is initiated. As $\boldsymbol{x}_{\text{start}}$ is the position of the misplaced image data in the texture unwrap and $\boldsymbol{x}_{\text{end}}$ is the position of the image data in the reference frame, correspondence of the relevant vertices in M_i can be established via Eq. (5). To achieve the transformation, the vertex positions V_i of mesh M_i are adjusted by solving a set of linear equations. The parameters to be found are again the offsets from the initial to the modified mesh vertices $\theta = \Delta\boldsymbol{v}$, as defined by the modification of the mesh. The adjustment term consists of a single equation for the two coordinate directions:

$$E_{\text{A}} = |\boldsymbol{B}(\boldsymbol{x}_{\text{end}})\theta - \Delta\boldsymbol{x}_{\text{src}}|$$

where $\boldsymbol{B}$ contains barycentric coordinates and the propagation of the adjustment is facilitated by applying the uniform Laplacian $\boldsymbol{L}$ as defined above. The radius of influence is modeled using a damping identity matrix scaled by an inverse Gaussian G whose standard deviation is set according to the influence radius r. With these three terms, the new vertex positions can be obtained by solving:

$$\hat{\theta} = \underset{\theta}{\text{argmin}}\,(E_{\text{A}}(\theta) + \lambda_{\text{L}} E_{\text{L}}(\theta) + G\theta)$$

These $4|V| + 2$ linear equations are independent of the image data and the equations, and can be solved in real time. Note that the equations in x and y are independent of each other and can be solved separately.

The main benefit of using this tool in texture space is that assessment and interaction can be performed locally. Only small cursor movements are required to correct erroneous drift and iterative fine tuning can easily be performed in combination with the tools shown in Fig. 2.

4.4 Correspondence tool

The correspondence tool lets a user mark the location of a distinct point $\boldsymbol{x}_{\text{ref}} \in R_0$ inside the reference region (the white points in Figs. 2 and 4). As mentioned above, the visualization of this location stays static in texture space; it has proven to be a very powerful assessment tool. A correspondence is established by marking the correct position $\boldsymbol{x}_{\text{cur}}$ of the feature in the texture unwrap of the current frame I_i (the green point in Fig. 4). Translated to the adjustment tool, $\boldsymbol{x}_{\text{cur}}$ is the data found in a wrong location (i.e., $\boldsymbol{x}_{\text{start}}$) and $\boldsymbol{x}_{\text{ref}}$ is the position where it should be moved to (i.e., $\boldsymbol{x}_{\text{end}}$). It should be noted that $\boldsymbol{x}_{\text{cur}}$ marks the location of image data inside I_i, rather than a position in texture space. So whenever M_i changes for any reason, the location of $\boldsymbol{x}_{\text{cur}}$ has to be adapted. An arbitrary number of correspondences between the reference frame and the current frame can be set. To avoid confusion, the visualizations of corresponding points are connected by a green line. Note that as the sparse correspondences represent static image locations and are therefore independent of tracking results, they can also be derived from an external source, e.g.,

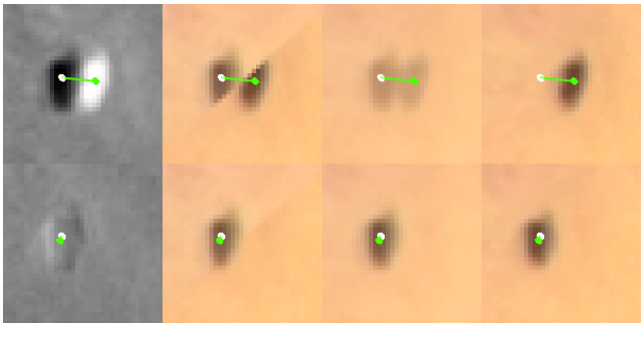

Fig. 4 Correspondence tool. Top: tool applied to the first row of Fig. 2. Bottom: result of data-based optimization incorporating the correspondence.

by facilitating a point tracker in a host application. Naturally, they can only be visualized in texture space if tracking data is available.

The alignment based on those correspondences extends the adjustment term introduced for the adjustment tool to incorporate multiple equations for the correspondence vectors pointing from $\boldsymbol{x}_{\text{ref}}$ to $\boldsymbol{x}_{\text{cur}}$.

$$E_{\text{C}}(\theta) = \sum |\boldsymbol{B}(\boldsymbol{x}_{\text{ref}})\theta - \Delta\boldsymbol{x}_{\text{src}}|$$

Finding a purely geometric solution is again possible and can make sense for single frames containing very unreliable data (e.g., strong motion blur). However, in most cases a more elegant approach is to include the correspondences as additional constraints directly into the image data-based optimization. As the mesh changes in each iteration, the correspondence vectors have to be updated each time using Eq. (5). However, as mentioned above, the location $W_{0i}(\boldsymbol{x}_{\text{cur}})$ is constant in the current frame and is therefore only calculated before the first iteration based on the initial mesh M_i. The correspondence term E_{C} is added to the objective function defined in Eq. (3):

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \left(\lambda_{\text{D}} E_{\text{D}}(\theta) + \lambda_{\text{L}} E_{\text{L}}(\theta) + \lambda_{\text{C}} E_{\text{C}}(\theta)\right)$$

The parameter λ_{C} can be used to communicate the accuracy of the provided correspondence. For the results in Fig. 4, the confidence in this accuracy was set low, giving more relevance to the underlying data. This is reflected by the slight misalignment of the input points, but correct alignment of the data.

The main benefit of applying this tool in texture space is again that assessment and interaction can be performed locally. Communicating drift by specifying the location of content deviating from a reference location has proven to be a very natural process that only requires small cursor movements.

4.5 Influence and smoothness brushes

The influence and smoothness brushes are both painting tools that allow the user to specify characteristics of image regions in the sequence. The influence brush facilitates a per-pixel (i.e., per-equation) scaling λ_{D_k} of the data term while the smoothness brush represents a per-vertex (i.e., per-equation) scaling λ_{L_k} of the Laplacian term. In both cases this can be seen as an amplification (> 1) or weakening (> 0 and < 1) of the respective λ parameter for the specific equation. Visualization is based on a simple color scheme: green stands for amplification, magenta represents weakening, and the transparency determines the magnitude. For users who prefer to create the weights independent of the plugin, e.g., by using tools inside the host applications, an interface for influence and smoothness maps containing the values for λ_{D_k} and λ_{L_k} respectively is provided.

The main application of the influence brush is to weaken the influence of data in subregions that are very unreliable or erroneous. This can be a surface characteristic, e.g., the blinking eye in Fig. 5, or a temporary external disturbance like occluding objects or reflections. The smoothness brush can be used to model varying surface characteristics or to amplify regularization for low texture areas. A typical application for varying dynamics is for the bones and joints of an articulated object.

If actual surface properties are to be modeled or an expected disturbance occurs in the same part of the surface throughout the sequence, those characteristics can be set in I_0 and can be propagated throughout the tracking process. The idea behind the propagation is that a "verified" tracking result exists upto the frame previous to the currently processed one. Therefore, mapping the brush information from the reference frame to the previous one naturally propagates the previously determined surface characteristics to the correct location. Figure 5 illustrates how the influence brush can be applied in texture space to tackle a surface disturbance caused by a blinking eye.

For occluding objects, vanishing surfaces, or temporary disturbances (e.g., motion blur or highlights), the brushes can be set for individual frames. Generally, propagation does not work for these use cases since the disturbance is not bound to the surface. However, in texture space the actual motion of a disturbance is usually very restricted. Therefore, propagation in combination

Fig. 5 Influence brush. Left to right: the reference region, an erroneous result, application of the influence brush to weaken the influence of the image data, and the corrected frame.

with slight user adjustments creates a very efficient workflow. Established brush tools in combination with keyframing or even tracking of the overlaying object in the host application can also be used for this kind of correction.

5 Results

In a first experiment, we evaluated the capability of the proposed tools to correct tracking errors caused by large frame-to-frame motion. To do so, we increased the displacements by dropping frames from the 720×480 pixel *Face Capture* sequence [24], originally designed for post-production students to master their skills on material that is representative of real world challenges. While the original sequence was tracked correctly, tracking breaks down at displacements of around 50 pixels. Figure 6 illustrates for one frame how the correspondence tool can be used to correct such tracking errors with minimal intervention. In this example, our automatic approach using default parameters can track from frame 1 directly to frames 2–12. However, trying to directly track to frame 13 fails. A single manual approximate correspondence provided by the user effectively solves the problem.

The remaining results in this section were created using production quality 4k footage that we are releasing as open test material alongside this publication. The *sailor* sequence depicted in Fig. 7 shows the flexing upper arm of a man. The post-production task we defined was to stick a temporary tattoo onto the skin of the arm. A closeup of the effect is depicted below the samples. Note the strongly non-rigid deformation of the skin and therefore of the anchor overlay. Also note the change in shading on the skin; it is estimated and applied to the tattoo. The texture unwrap (without photometric compensation) in Fig. 7 highlights how the complex lighting and surface characteristics lead to very different appearances of the skin throughout the sequence. While geometric alignment and photometric properties were estimated fairly accurately, the shifted difference images depicted in Fig. 9 show a considerable texture differences.

Fig. 6 Correcting tracking errors caused by large displacements. Top left to bottom right: part of reference frame 1 with tracking region marked, frame 12 with tracking from reference still working, frame 13 with tracking directly from frame 1 failing, provision of a single correspondence as hint (green), correct tracking with additional hint, and estimated displacement vector for corrected point.

Fig. 7 Samples from the *sailor* sequence (100 frames) and a closeup of the same samples including the visual effects.

Fig. 8 Texture unwrap of the same samples as in Fig. 7.

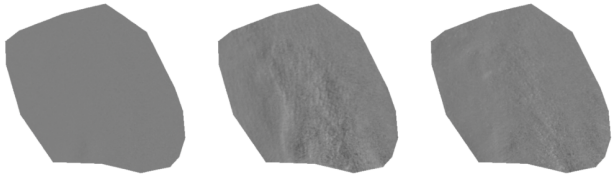

Fig. 9 Shifted differences of the unwrapped samples in Fig. 7 and the reference region.

The reason is that estimation of photometric parameters is limited to smooth, low frequency shading properties and cannot capture fine details like shadows of the bulging skin pores in this example. However, on playing back the unwrapped *sailor* sequence, it can be observed that the overall surface stays fairly static, suggesting that the tracking result is adequate. Nevertheless, there is a distinct disturbance for a few frames in a confined region of the image. Figure 10 shows from left to right the reference image, the unwrap just before the disturbance starts, the unwrapped frame of maximum drift, and the corrected version. A drifting structure exists as a vertical dark ridge passing through the overlayed reference point. The ridge drifts about 5 pixels to the right. As this feature cannot be distinguished in the reference image, we use an adjacent frame as reference for the correction. The issue can be solved with both the smoothness brush and the adjustment tool. Applying the smoothness brush is particularly easy, as the problematic region is very confined and can just be covered by a small static matte in texture space. The adjustment tool can also easily be applied in a single frame in texture space and the corrected result can be propagated to eliminate the drift in all affected frames. As there is considerable global motion in the sequence and the issue is very confined and subtle, we found that assessment and interaction in texture space is the only effective and efficient way to detect, quantify, and solve the problem.

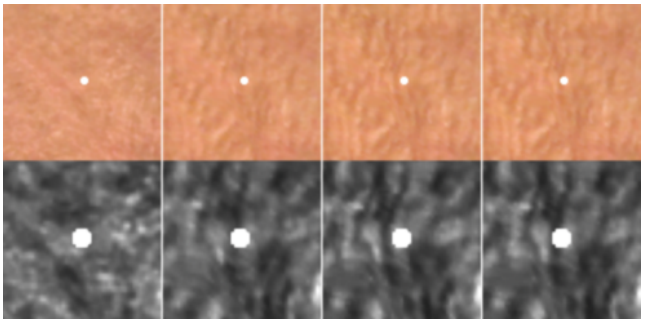

Fig. 10 Problematic region in texture space and high contrast closeup for better assessment.

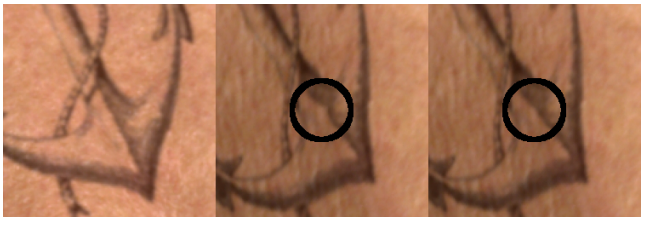

Fig. 11 Closeup of a problematic tracking region in the final composite.

The *wife* sequence depicted in Fig. 12 shows a woman lifting her head and wiping hair out of her face. The post-production task we defined was to age her by painting wrinkles on her face. The final effect is depicted below the samples. Note the opening of the mouth and eyes, the occlusion by the arm, and the change in facial expression. Good initial results can be achieved for the tracking of the skin. However, the opening of the mouth, the blinking of the eyes, and the motion of the arm create considerable problems. Due to the confinement of the disturbance, both the influence and the smoothness brush can be applied for the mouth and the eyes. See Fig. 5 for a similar use-case. In this specific case, an adjustment of the global smoothness parameter λ_L adequately solved the issue. One distinct problem to be solved is the major occlusion by the wife's arm where she is wiping the hair out of her face. To have an unoccluded reference texture, tracking was started at the last frame and performed backwards. Figure 13 highlights that while most of the sequence tracks perfectly well, at the end of the sequence major disturbances occur. To solve this problem, the influence brush is applied in texture space. For this, we used the built-in Roto tool in NUKE with only 5 keyframes. The resulting matte can be reused in compositing to limit the painted overlay to the surface of the face. Figure 14 illustrates application of the influence brush to two problematic frames.

Fig. 12 Samples from the *wife* sequence (100, 8, and 1) and the same samples including the visual effect.

Fig. 13 Unwrapped samples of the *wife* sequence (100, 50, and 1).

Note the improvements around the mouth and on the cheek.

In order to evaluate the effectiveness of the proposed tools and workflows, post-production companies compared the NUKE plugin tool with other existing commercial tools in real-world scenarios. Different usability criteria were rated on a 5-point scale and passed back together with additional comments. The resulting feedback showed that the proposed method was rated superior to the

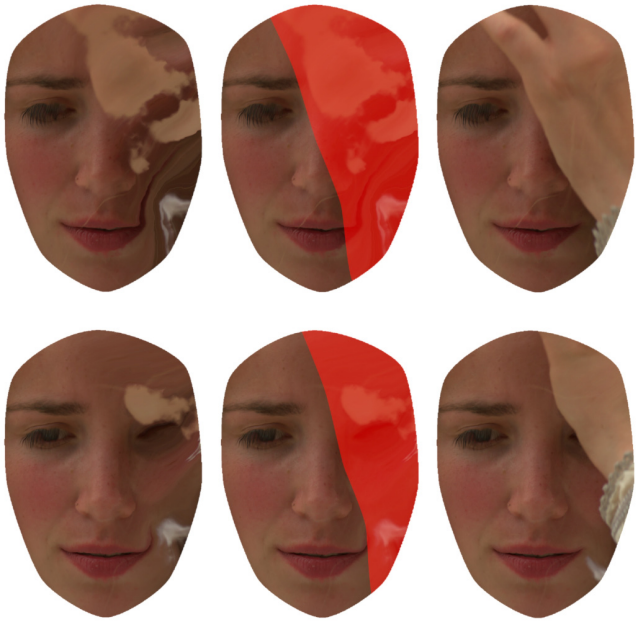

Fig. 14 Texture unwrap of samples 8 and 1 of the *wife* sequence, application of the influence brush in texture space, and resulting tracking improvement.

other tools. Most criteria were judged slightly better while "usefulness" and "overall satisfaction" were rated clearly higher, indicating that consideration of user hints in deformable tracking can enhance real visual effects workflows.

6 Conclusions

We have introduced a novel way of assessing and interacting with surface tracking results and algorithms based on unwrapping a sequence to texture space. To prove applicability to the relevant use-cases, we have implemented our approach as a plugin for an established post-production platform. Assessing the quality of tracking results in texture space is equivalent to detecting geometric (and photometric) changes in a played back sequence. We found that this is a simple task even for an untrained casual observer and that established post-production tools can help to pinpoint even minimal errors. Therefore, assessment has proven to be very effective. The application of user interaction tools directly in texture space in combination with iterative re-optimization of the result has proven to be intuitive and effective. The most striking benefits of applying tools in texture space is that interaction can be focused on a very localized area and that only small cursor movements are required to correct errors. We believe that there is a high potential in pursuing both research and development in texture space assessment and user interaction for tracking applications.

Acknowledgements

This work was partially funded by the German Science Foundation (Grant No. DFG EI524/2-1) and by the European Commission (Grant Nos. FP7-288238 SCENE and H2020-644629 AutoPost).

Electronic Supplementary Material Supplementary material is available in the online version of this article at http://dx.doi.org/10.1007/s41095-017-0089-1.

References

[1] Imagineer Systems. mocha Pro. 2016. Available at http://www.imagineersystems.com/products/mocha-pro.

[2] Foundry. NUKE. 2016. Available at https://www.foundry.com/products/nuke.

[3] The Pixelfarm. PFTrack. 2016. Available at http://www.thepixelfarm.co.uk/pftrack/.

[4] Klose, F.; Ruhl, K.; Lipski, C.; Magnor, M. Flowlab—An interactive tool for editing dense image correspondences. In: Proceedings of the Conference for Visual Media Production, 59–66, 2011.

[5] Ruhl, K.; Eisemann, M.; Hilsmann, A.; Eisert, P.; Magnor, M. Interactive scene flow editing for improved image-based rendering and virtual spacetime navigation. In: Proceedings of the 23rd ACM International Conference on Multimedia, 631–640, 2015.

[6] Zhang, C.; Price, B.; Cohen, S.; Yang, R. Highquality stereo video matching via user interaction and space–time propagation. In: Proceedings of the International Conference on 3D Vision, 71–78, 2013.

[7] Re:Vision Effects. Twixtor. 2016. Available at http://revisionfx.com/products/twixtor/.

[8] Wilkes, L. The role of ocula in stereo post production. Technical Report. The Foundry, 2009.

[9] Chrysos, G. G.; Antonakos, E.; Zafeiriou, S.; Snape, P. Offline deformable face tracking in arbitrary videos. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, 1–9, 2015.

[10] Rother, C.; Kolmogorov, V.; Blake, A. "GrabCut": Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics* Vol. 23, No. 3, 309–314, 2004.

[11] Liao, M.; Gao, J.; Yang, R.; Gong, M. Video stereolization: Combining motion analysis with user interaction. *IEEE Transactions on Visualization & Computer Graphics* Vol. 18, No. 7, 1079–1088, 2012.

[12] Wang, O.; Lang, M.; Frei, M.; Hornung, A.; Smolic, A.; Gross, M. Stereobrush: Interactive 2D to 3D conversion using discontinuous warps. In: Proceedings of the 8th Eurographics Symposium on Sketch-Based Interfaces and Modeling, 47–54, 2011.

[13] Doron, Y.; Campbell, N. D. F.; Starck, J.; Kautz, J. User directed multi-view-stereo. In: *Computer Vision–ACCV 2014 Workshops*. Jawahar, C.; Shan, S. Eds. Springer Cham, 299–313, 2014.

[14] Bartoli, A.; Zisserman, A. Direct estimation of non-rigid registrations. In: Proceedings of the 15th British Machine Vision Conference, Vol. 2, 899–908, 2004.

[15] Zhu, J.; Van Gool, L.; Hoi, S. C. H. Unsupervised face alignment by robust nonrigid mapping. In: Proceedings of the IEEE 12th International Conference on Computer Vision, 1265–1272, 2009.

[16] Hilsmann, A.; Eisert, P. Joint estimation of deformable motion and photometric parameters in single view videos. In: Proceedings of the IEEE 12th International Conference on Computer Vision Workshops, 390–397, 2009.

[17] Gay-Bellile, V.; Bartoli, A.; Sayd, P. Direct estimation of nonrigid registrations with image-based self-occlusion reasoning. *IEEE Transactions on Pattern Analysis & Machine Intelligence* Vol. 32, No. 1, 87–104, 2010.

[18] Seibold, C.; Hilsmann, A.; Eisert, P. Model-based motion blur estimation for the improvement of motion tracking. *Computer Vision and Image Understanding* DOI: 10.1016/j.cviu.2017.03.005, 2017.
[19] Hilsmann, A.; Schneider, D. C.; Eisert, P. Image-based tracking of deformable surfaces. In: *Object Tracking*. InTech, 245–266, 2011.
[20] Pilet, J.; Lepetit, V.; Fua, P. Real-time nonrigid surface detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 822–828, 2005.
[21] Brox, T.; Malik, J. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 33, No. 3, 500–513, 2011.
[22] Wedel, A.; Cremers, D.; Pock, T.; Bischof, H. Structure- and motion-adaptive regularization for high accuracy optic flow. In: Proceedings of the IEEE 12th International Conference on Computer Vision, 1663–1668, 2009.
[23] Brox, T.; Bruhn, A.; Papenberg, N.; Weickert, J. High accuracy optical flow estimation based on a theory for warping. In: *Computer Vision–ECCV 2004*. Pajdla, T.; Matas, J. Eds. Springer Berlin Heidelberg, 25–36, 2004.
[24] Hollywood Camera Work. Face Capture dataset. 2016. Available at https://www.hollywoodcamerawork.com/tracking-plates.html.

Johannes Furch is a research associate at Fraunhofer HHI. He received his diploma in computer science from the University of Tübingen in 2010. His research focuses on video-based motion analysis for application in visual media production.

Anna Hilsmann received her Dipl.-Ing. degree in electrical engineering and information technology from RWTH Aachen in 2006 and her Dr.-Ing. degree in computer science from HU Berlin in 2014. She joined the Computer Vision and Graphics Group at Fraunhofer HHI in 2007 and the Visual Computing Group at HU Berlin in 2011. Since 2015, she has headed the Computer Vision and Graphics Group at Fraunhofer HHI. Her main research interests cover 3D image and video analysis, such as image registration, model-based deformable tracking and 3D reconstruction, as well as synthesis, image- and video-based rendering, animation, and editing.

Peter Eisert is professor for visual computing at Humboldt University, Berlin and heads the Vision & Imaging Technologies Department of the Fraunhofer HHI, Berlin, Germany. He received his Dipl.-Ing. degree in electrical engineering from TU Karlsruhe and his Dr.-Ing. degree from the University of Erlangen, Germany. In 2001, he worked as a postdoctoral fellow at Stanford University on 3D image analysis and synthesis as well as facial animation and computer graphics. He joined HHI in 2002 and HU Berlin in 2009, where he is coordinating and initiating numerous national and international research projects. He has published more than 150 conference and journal papers and is the associate editor of the *International Journal of Image and Video Processing* as well as on the editorial board of the *Journal of Visual Communication and Image Representation*. His research interests include 3D image analysis and synthesis, face processing, image-based rendering, computer vision, and computer graphics.