Visual Comput (2006)
DOI 10.1007/s00371-006-0050-2

ORIGINAL ARTICLE

Darko Pavić
Volker Schönefeld
Leif Kobbelt

Interactive image completion with perspective correction

D. Pavić (✉) · V. Schönefeld · L. Kobbelt
Computer Graphics Group, RWTH Aachen University, Germany
{pavic, kobbelt}@cs.rwth-aachen.de

Abstract We present an interactive system for fragment-based image completion which exploits information about the approximate 3D structure in a scene in order to estimate and apply perspective corrections when copying a source fragment to a target position. Even though implicit 3D information is used, the interaction is strictly 2D, which makes the user interface very simple and intuitive. We propose different interaction metaphors in our system for providing 3D information interactively. Our search and matching procedure is done in the Fourier domain, and hence it is very fast and it allows us to use large fragments and multiple source images with high resolution while still obtaining interactive response times. Our image completion technique also takes user-specified structure information into account where we generalize the concept of feature curves to arbitrary sets of feature pixels. We demonstrate our technique on a number of difficult completion tasks.

Keywords Image completion · Image repair · Example-based synthesis · User interface

1 Introduction

Fragment-based image completion techniques are a very powerful tool to fill in missing pixel information, e.g., when removing a foreground object from a digital photo. The conceptual idea is to fill a hole in the image by copying small *source* fragments from known regions of the image such that they eventually completely cover the undefined region. The mutual overlap of the *target* fragments and the overlap between target fragments and the boundary of the hole is used to compute a similarity measure which controls the selection of the best source fragment candidate in order to guarantee a seamless appearance of the completed image.

The various approaches to fragment-based image completion mainly differ in three aspects. First, in the definition of the *search space*. Source fragments can be taken anywhere from the source image or only from certain user-defined sub-regions in order to preserve specific feature information or to reduce time complexity. Second, the selection of the best source fragment is based on a *similarity measure*, which can use pixel color information as well as structural information such as the presence and orientation of image features. Third, once the best source fragment is found, it has to be *transformed* to the target location in the image. Besides mere translation, certain types of affine transforms, like scaling and rotation, have been proposed.

The fundamental assumption, which justifies the fragment-based image completion approach, is that for a small enough image fragment, we can assume the scene, which is visible in this fragment, to be planar. Hence we can ignore all kinds of occlusion and dis-occlusion effects when copying a fragment from one image location to another, and therefore we do not need any true 3D information about the scene. However, the restriction to affine transforms of the fragments, as it has been done in the previous work, mathematically corresponds to the even more strict and somewhat unrealistic assumption that these planar scene fragments are aligned to the image plane. A natu-

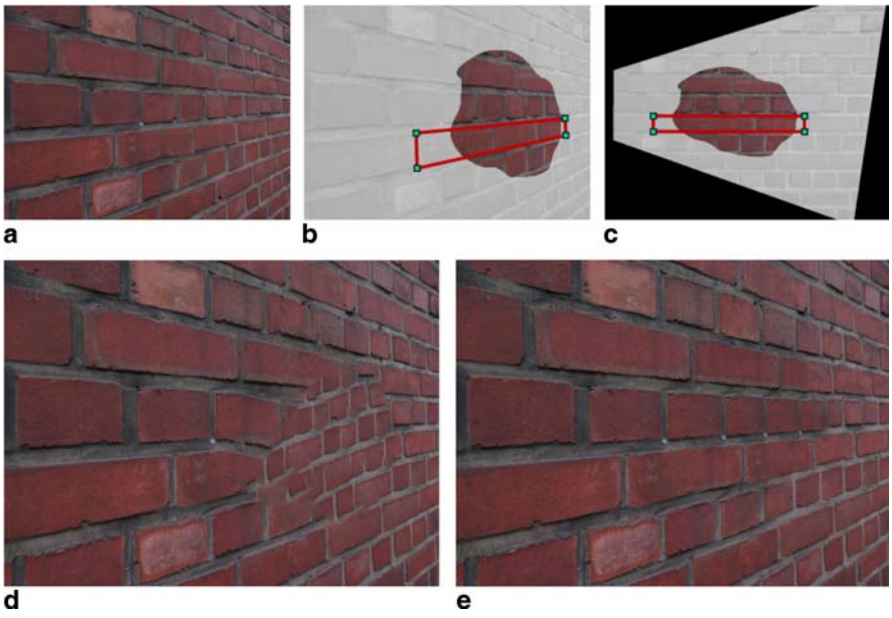

Fig. 1a–e. The effect of perspective corrections for image completion. **a** shows the input image. **d** shows the completion without using perspective correction which leads to strong perspective artifacts. When using the rectifying homography defined in **b** and then performing the completion in the rectified space **c** we obtain the solution in **e**

ral generalization of the existing approaches is, therefore, to allow for projective transforms of the image fragments, which enables the compensation of perspective distortion when the source and target scene fragments are not lying in the same supporting 3D plane.

In this paper we present a new system for interactive image completion that applies perspective corrections when copying fragments. Through a simple interaction metaphor the user can define a set of projective transforms. Based on this information the system rectifies the corresponding image regions and then performs the image completion in rectified image space. An intuitive extension of this user interface allows the system to handle even more complex scene geometries such as moderately curved surfaces. An automatic snapping mechanism for the points selected by the user guarantees continuity at the boundaries of adjacent rectified regions. User-defined feature information is taken into account by encoding feature proximity as an additional color channel to the image.

Figure 1 shows a simple example for an application of our interaction metaphor and the resulting image completion which is free from perspective artifacts. In the results Sect. 9 we present more examples, which demonstrate the power and usefulness of our system.

2 Related work

There are two fundamental approaches to image completion: *image inpainting methods* and *example-based approaches*. Image inpainting [3, 5, 6, 28] is good in filling small missing areas like thin gaps, e.g., when removing scratches from old photographs, but the diffusion process of image inpainting leads to blurring artifacts when trying to complete large missing areas, e.g., after removing large foreground objects from images.

Example-based approaches have their origin in texture synthesis, where large texture patches are synthesized from small texture samples on a per-pixel basis using a pixel neighborhood for color prediction [2, 14, 32]. Instead of this per-pixel procedure some approaches use larger patches or fragments [13, 25] in order to speed up the process. By using graph-cut techniques [24], optimal boundaries between patches can be computed.

In the context of image completion there are some approaches working on a per-pixel basis [7, 16, 20], but fragment-based approaches [4, 9, 12, 31] usually produce superior results. The automated method presented by Drori et al. [12] leads to very impressive completions but at the cost of high computation time. Our method achieves results of comparable quality but it is up to two orders of magnitude faster allowing for interactive control of the process.

An important observation for image completion is that the propagation of structure should be treated separately from texture propagation [7, 9]. Recently Sun et al. [31] presented an interactive approach, where feature curves for structure propagation are supplied by the user. Putting a human user into the loop usually leads to much more plausible completions than the ones generated by sophisticated heuristics. Hence user-control is considered an important feature. In our system the user also has the possibility to specify structure information as additional color channel to the image.

Other previous completion techniques take user-defined *points of interest* [12] into account to control the propagation direction when completing rotationally symmetric

shapes. Depth images have also been used to restrict the search for source fragments to regions of similar depth in order to avoid artifacts emerging from perspective distortion [16, 30]. Wilczkowiak et al. [33] have mentioned perspective correction but not really explored it in the context of image hole filling. In this paper the perspective correction is investigated much more thoroughly and explained in detail so that it can be reproduced easily. Moreover, we embed it into an easy-to-use interactive workflow and generalize it to continuous piecewise homographies, which enables the rectification of more complex scene geometries.

There are also a number of papers where non-greedy methods are used. Kwatra et al. [23] propose a global texture optimization and very recently Komodakis and Tziritas [22] proposed a global optimization method for image completion. However they also do not consider the problem of perspective distortion and show only low-resolution examples.

Our approach is also somewhat related to methods from the field of image-based modeling and editing [11, 19, 27], where user interaction to supply 3D information about the underlying scene is a very important part of the workflow. Oh et al. [27] represent a scene as a layered collection of depth images, which are created manually. Horry et al. [19] generate animations from single images by providing perspective information through a vanishing point and a spidery mesh. Debevec et al. [11] describe hybrid approach combining image-based and geometry-based modeling metaphors for the reconstruction of architecture.

Drawing quads to convey perspective information has been used in other image processing applications before. But embedding this metaphor as well as the quad-grid metaphor in the interactive image completion workflow is new. Liu et al. [26] use the quad-grid metaphor to manually tag the periodicity of a non-planar texture sample and typically several repetitions of the underlying pattern have to be selected. In our case, we use it to define piecewise homographies for the rectification of non-planar surfaces. The technical contribution lies in a relaxation procedure to ensure continuity between the homographies, i.e., to ensure that there are no jumps between neighboring facets. Without this snapping technique the metaphor would not be practical since the user would have to click image positions to sub-pixel precision.

3 Overall system description

Our interactive system uses several image buffers as *source buffers* $S(\cdot,\cdot)$. When searching for fragments, one buffer is set as the *target buffer* $T(\cdot,\cdot)$. Since all buffers are treated the same, our system explicitly supports scenarios where an image is completed by transforming source fragments from several other input images.

As already stated in the introduction, fragment-based image completion techniques make the assumption that the scene is locally (within one fragment) flat such that copying fragments becomes essentially a 2D operation. The restriction to affine transformations of fragments (shifting, scaling, rotation) as done in previous approaches, further assumes that all planar fragments are aligned to the image plane. This, however, is not the case in most scenes.

We, therefore, generalize previous approaches by taking the estimated 3D orientation of the flat fragments into account. The basic idea is to *rectify* image regions by applying projective transforms to the input image(s). The term rectification refers to a 2D transform which aligns arbitrary 3D planes to the image plane. To copy a fragment from one rectified image to another then corresponds to applying a projective transform to the fragment.

The workflow of our system is as follows: After loading the input image(s) the user first paints an α-matte which defines the image regions to be replaced ($\alpha = 1$). Then he can define a set of 3D planes in the image which are used for rectification. Each rectification generates another image buffer where we can later search for source fragments. If no 3D plane is specified, our system behaves just like previous image completion techniques without perspective corrections. Optionally, the user can further define a set of features (lines, curves, or arbitrary sets of pixels) which will be used to restrict the source fragment search in order to preserve these features.

After the specification phase, the user interactively picks target fragment locations (usually near the boundary of the undefined region) and chooses target fragment sizes, and the system finds the best fitting source fragment. By this the unknown region ($\alpha = 1$) is incrementally filled with fragments. Since we use a FFT-based approach for the searching procedure, the response times of our system are fractions of a second even if we search for large fragments in high-resolution source buffers.

Notice that even if we are implicitly using 3D information, the user interface is completely 2D. The user only draws curves or picks locations on 2D images. In the following sections we will explain which intuitive interaction metaphors make this possible.

4 Image completion with perspective correction

Since the input image and the rectified source buffers are related by known perspective transforms, we can apply the same transformations to the α-matte and hence the roles of source and target buffers can be exchanged in the image completion phase.

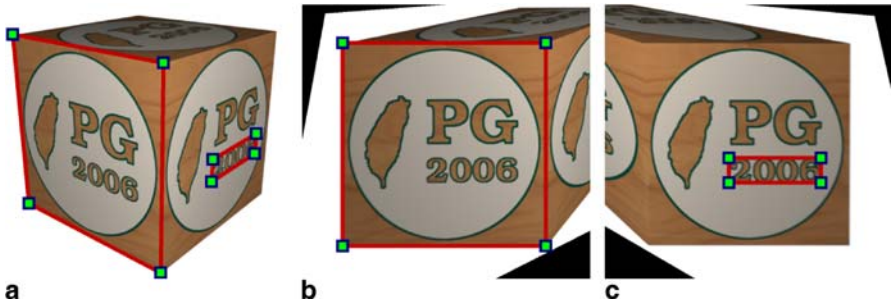

Fig. 2a–c. Homographies (or 3D planes) are defined by drawing convex quads directly in the image (**a**). These quads are mapped to rectangles with a user-specified aspect ratio (**b**, **c**). Each rectified image serves as an additional image buffer. The quads do not have to bound the rectified region as shown in **c**

Due to their rotational invariance, we prefer to use circular fragments. When applying a projective transform, such a fragment would be mapped to an ellipse which would make fragment matching computationally more involved (and less intuitive). However, we can exploit the freedom to switch the target buffer and to use an individual source buffer for each rectified region in order to avoid the use of elliptical fragments.

Assume we had a circular planar fragment in 3D object space (e.g., one of the circles on the sides of the cube in Fig. 2) which is mapped to an ellipse in image space (the input image (a)). Rectifying the supporting plane (the side of the cube) corresponds to aligning the fragment to the camera plane such that it becomes circular in the distorted image (subfigures (b) and (c)). Hence if we restrict the selection of target fragments to the rectified region of the target buffer and the source fragment search to the rectified regions of the source buffers then circular fragments are always mapped to circular fragment since they, in fact, correspond to circular fragments in 3D object space. The perspective distortion of the fragment then becomes effective only when the completed region is mapped back to the input image by un-rectifying the target buffer.

In practice it turns out that we do not even have to explicitly restrict the fragment search to the rectified regions since the most similar fragment is usually found in this region anyway. Due to the fact that we are using a FFT-based evaluation procedure, the unsuccessful search in the distorted regions is faster than explicitly checking whether a given source fragment belongs to the rectified region. Moreover, as shown in the Colosseum example in Fig. 9, it is not needed to rectify the whole missing region explicitly.

5 Sketching a homography

While the surface of real objects in a 3D scene can often be assumed to be locally planar, their surface orientation is usually not aligned to the image plane of the camera. Hence when copying fragments we have to compensate the perspective distortion by applying a projective transformation. Notice that even if we implicitly exploit 3D

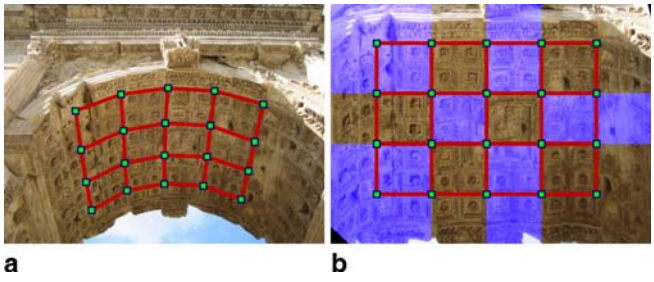

Fig. 3a,b. Quad-Grid interaction metaphor. Curved surfaces can be rectified by defining a regular quad grid as shown in **a**. Rectifying each quad and putting the resulting rectangles together yields an unfolded orthogonal view of the surface as visualized in **b**

information for this operation, a projective transformation of a plane is essentially a 2D transform, a *homography*, which can be represented by a 3×3 matrix in homogeneous coordinates.

The most natural way to define a homography is through a pair of quadrilaterals in image space. The four 2D point correspondences provide the eight constraints by which the nine entries of the 3×3 matrix can be computed up to a constant factor (which establishes uniqueness in homogeneous coordinates) [17].

In our system the user does not want to specify an arbitrary homography but a homography which is supposed to rectify an image region whose 3D pre-image is sufficiently planar. Hence it suffices that the user defines just one convex but otherwise general quad while the second quad is restricted to be a rectangle where the user only has to adjust the aspect ratio. This leads to a very intuitive user interface where the user simply draws a quad directly onto the image. The shape of the quad should be such that it covers an image region which corresponds to a rectangular 3D pre-image. Figure 2 shows an example of this interaction.

Notice that since the fragment search is always applied to the full image buffer, there is no need to use the homography defining quad as a clipping frame and to consider only the interior of the quad as the rectified region. In practice it is very unlikely that a matching circular fragment is found in the non-rectified region (region not parallel to the image plane after the rectification) and hence we do not

need to explicitly segment the input image into planar regions. This observation is extremely helpful since we can, e.g., use a single window of a building to define the homography for the entire facade.

6 More complex interaction metaphors

When dealing with more complex scenes that contain even curved surfaces like cylinders it can become tedious to manually define all the homographies for each small part of the surface. Hence we implement another interaction metaphor which allows us to define a whole grid of quads $Q_{i,j}$ in an efficient manner. Each of the quads in the grid implies a homography $H_{i,j}$ which rectifies the interior of $Q_{i,j}$. Putting all these rectified pieces together yields an unfolded orthogonal view of the whole curved surface (see Fig. 3). Unlike Liu et al. [26] we do have not to cover exactly one texture element with each quad of the grid (see Fig. 4 or Colosseum example in Fig. 9).

The idea is to generate a regular quad grid by first selecting a polygon with n vertices and then offsetting this polygon to generate a $n \times m$ grid of vertices $\boldsymbol{p}_{i,j}$. Groups of vertices can be selected and shifted simultaneously to quickly align the quads to the perceived 3D orientations (see the accompanying video [29]).

While this metaphor allows even untrained users to quickly generate a rough approximation of a consistent quad mesh, it turns out to be quite difficult to position the vertices with a sufficient precision such that the rectified image is a *continuous* deformation of the input image (see Fig. 4). The reason for this difficulty is that even if the homographies $H_{i,j}$ are defined by the common corners $\{\boldsymbol{p}_{i,j}, \boldsymbol{p}_{i+1,j}, \boldsymbol{p}_{i,j+1}, \boldsymbol{p}_{i+1,j+1}\}$ of the quads $Q_{i,j}$ there is no guarantee that the homographies of two neighboring quads, e.g., $H_{i,j-1}$ and $H_{i,j}$ coincide on the whole common boundary edge $\overline{\boldsymbol{p}_{i,j}\,\boldsymbol{p}_{i+1,j}}$. Geometrically this is obvious since for an arbitrary pair of 2D quads there does not have to exist a spatial configuration of two 3D rectangles and a perspective mapping which projects the 3D rectangles to the 2D quads.

To make the quad grid metaphor less sensitive, we derive a snapping mechanism which drags the grid defined by the user to a nearby configuration satisfying the property that homographies belonging to neighboring quads are continuous along the common boundary edge.

For simplicity let us assume that the rectified quads are squares, i.e., $H_{i,j}(Q_{i,j}) = [i, i+1] \times [j, j+1]$. From projective geometry we know that two homographies are identical along a line if they agree on at least three points along that line [8]. Since by construction neighboring homographies $H_{i,j-1}$ and $H_{i,j}$ coincide at the common corner points $\boldsymbol{p}_{i,j}$ and $\boldsymbol{p}_{i+1,j}$, we can measure the inconsistency by looking at how strongly the two mappings deviate at the mid-point of the common edge (see Fig. 5).

Because our snapping mechanism adjusts the positions of the grid vertices $\boldsymbol{p}_{i,j}$, it is better to compute the deviation in image space and to consider the inverse homographies $H_{i,j}^{-1}$ which map unit squares $[i, i+1] \times [j, j+1]$ to image quads $Q_{i,j}$. For these the mid-point deviation on the edge $\overline{\boldsymbol{p}_{i,j}\,\boldsymbol{p}_{i+1,j}}$ is computed by $\|H_{i,j-1}^{-1}(i+\frac{1}{2}, j) - H_{i,j}^{-1}(i+\frac{1}{2}, j)\|$. For a given grid point $\boldsymbol{p}_{i,j}$ we can sum up these mid-point deviations for all adjacent edges $\overline{\boldsymbol{p}_{i,j}\,\boldsymbol{p}_{i\pm1,j\pm1}}$ which gives us a quality score for this vertex.

Our grid snapping procedure is now a simple iterative relaxation procedure which moves each vertex in a direction which reduces its quality score. Since the procedure is quite fast there is no need for aggressive optimization. Hence we simply check for each vertex if moving it in one of the 8 principal directions improves the local quality and iterate over all interior vertices (Fig. 5). The boundary vertices are kept fixed to impose proper boundary conditions. Even if there is no strict mathematical guarantee for the convergence of this non-linear optimization, in practice the snapping procedure always worked robustly and quickly converged after only a few iterations.

Our quad-grid metaphor is much more flexible than letting the user specify a cylinder mapping since we can handle more complex non-planar configurations (see e.g., Fig. 9 top right). Also, defining a cylinder requires placing an arbitrarily oriented ellipse on the image, which is a more involved interaction than just clicking a number of points.

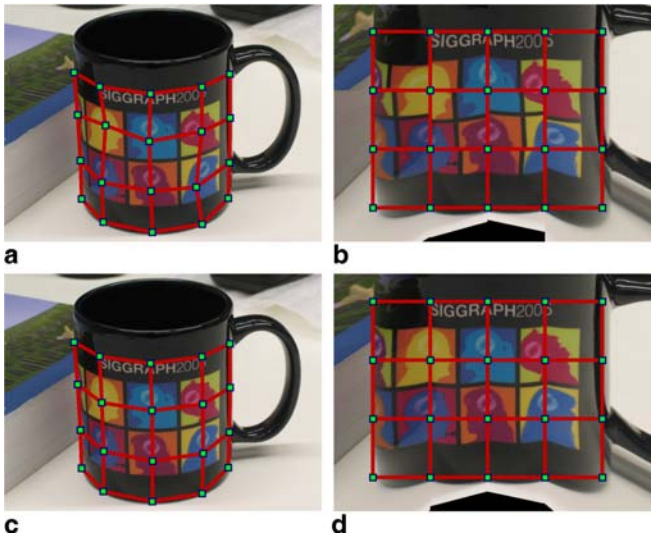

Fig. 4a–d. Quad-Grid relaxation. **a** shows the initial constellation of the quad-grid points. The piecewise homographies cause discontinuities at the common boundaries (**b**). **c** visualizes the same constellation after our relaxation procedure which eliminates the discontinuities as shown in **d**. The remaining distortions in the interior of the quads are due to the piecewise linear approximation of the cylindrical surface

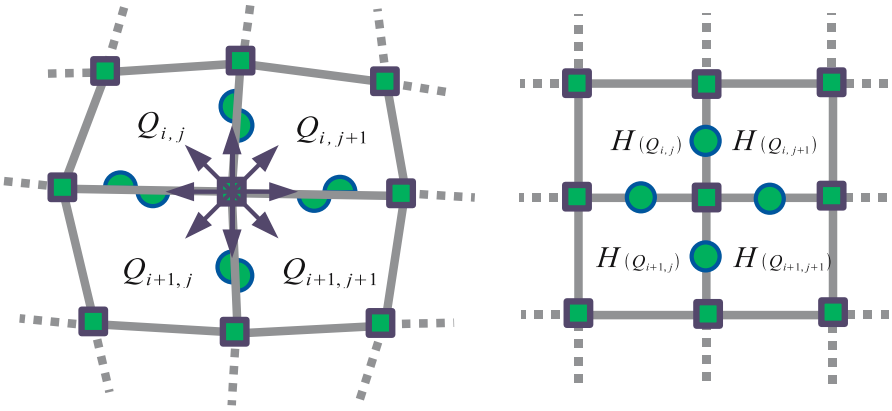

Fig. 5. Quad-Grid snapping. Midpoints of neighboring squares in rectified space (circles in the right image) are not necessarily mapped to the same point in image space (depicted by the half-circles on the left). Our snapping technique searches for the best update of each grid-point $\boldsymbol{p}_{i,j}$ in order to minimize the sum of midpoint deviations

7 Feature handling

As recently suggested by Sun et al. [31], feature preservation is an important aspect in image completion. Instead of adapting their technique, which uses continuous feature curves, we use an alternative pixel-based approach which nicely fits into our FFT-based computation (see Sect. 8) without causing significant overhead.

In our setting a feature is an arbitrary set of pixels that can even consist of several connected components. To define a feature, the user simply draws lines or free-hand curves onto the image or paints complete regions (see Fig. 9).

For each feature we compute a distance map where every pixel stores its Euclidian distance to the nearest feature pixel. We set the distance map resolution to the image buffer resolution and apply the homographies to the features as well. During the source fragment search, we take the feature information into account by treating it as an additional (color) channel of the buffer.

8 Implementation

We define colors as vectors in the CIE $L^*a^*b^*$ (short Lab) color space because it is adapted to human color perception [21]. The values for each *Lab*-channel are scaled *uniformly* to a subset of the unit cube $[0, 1]^3$.

When searching for source fragments, we apply the usual sum of squared distances (SSD) measure. Using similar ideas to those proposed by Kwatra et al. [24], we can exploit FFT in order to speed up the whole process by one to two orders of magnitude. Hel-Or et al. [18] propose a kernel projection technique for pattern matching based on the SSD measure which has an *expected* performance similar to our Fourier method. However, in real photographs and with larger fragments the situation often occurs that several source fragment candidates have quite similar matching scores and hence a relatively high number of kernel projections is necessary. This has a negative effect on the average performance. Furthermore the fast feedback of our system did not make the usage of methods necessary where nearest neighbors search is approximated [1, 10].

Checking fragment validity. For simplicity, we consider a source fragment as valid, if it lies completely in the interior of the source buffer (i.e., it lies completely within one period of the periodic source buffer function) and does not overlap with the unknown image region. The first validity check is trivially satisfied by restricting the source fragments to a region which is sufficiently far away from the source buffer boundary. For circular fragments with radius r we have to stay $2r$ pixels away from the lower and right boundaries (see rectangle in Fig. 6).

The second validity check can be implemented as another convolution operation and hence FFT can be exploited again. We take the α matte of the source buffer as the first buffer with $\alpha(u, v) = 1$ in undefined regions and zero everywhere else. For the second buffer we set $\beta(u, v) = 1$ in all pixels which belong to the interior of the flipped and padded target fragment and zero everywhere else. If the two buffers $\alpha(\cdot, \cdot)$ and $\beta(\cdot, \cdot)$ are convolved then non-zero entries in the result indicate source fragment positions where the fragment overlaps the undefined region. Hence valid fragments are those which correspond to a zero entry (see Fig. 6).

Feature handling. For each feature defined by the user, the image buffers are augmented by an additional buffer $D(\cdot, \cdot)$, which stores the distance map of that feature. Since features reach into the known as well as the unknown regions, the distance map $D(\cdot, \cdot)$ is defined everywhere in the image (including the target fragment). Features can be arbitrary sets of pixels and we do not impose any topological constraints. In order to evaluate the two-sided feature distance measure proposed by Sun et al. [31], we have to integrate the distance of all target feature pixels to the source feature and vice versa. Since the distance values are already stored in the distance map $D(\cdot, \cdot)$, we

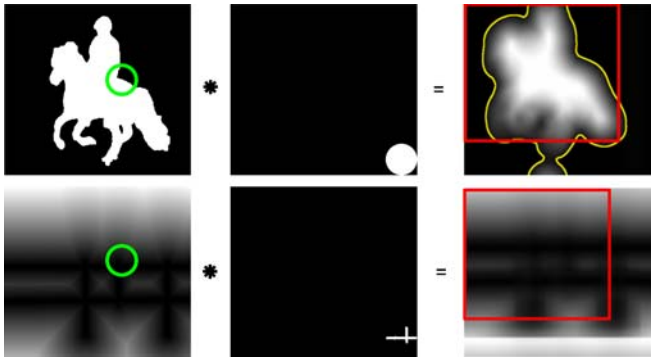

Fig. 6. The upper row shows the validity check for source fragments as a convolution operation. Convolving the full fragment mask with the α-matte leads to an image where valid fragments are indicated by pixels with value zero (the boundary of this region is depicted by the yellow line). The lower row shows how the one-sided feature distance is computed by convolving the fragment feature mask with the feature distance map of the buffer

can simply formulate the integration as another convolution.

From the distance map of the feature channel of the target fragment we extract the binary feature pixel mask $X_{(u,v)}(\cdot,\cdot)$ by thresholding:

$$X_{(u,v)}(u',v') = \begin{cases} 1 & D(u+u', v+v') \leq \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

After flipping and padding this binary mask, we obtain a buffer $X(\cdot,\cdot)$ whose convolution with $D(\cdot,\cdot)$ yields the one sided distances for each source fragment candidate (Fig. 6). The opposite distance is obtained analogously by thresholding the source buffer's feature channel and convolving it with the (flipped and padded) target fragment's feature distance map. Since both convolutions are computed in the Fourier domain, this operation can be computed quite efficiently.

For the normalization of the integrated distance values, we have to know the number of feature pixels in each fragment. This can be pre-computed by another convolution, this time by the thresholded feature map of the source buffer with the binary fragment mask $\beta(\cdot,\cdot)$, which we already used for the validity check.

Finally, in order to preserve features, we discard all source fragments whose two-sided feature distance measure (normalized by the number of feature pixels) is above a given tolerance h. This tolerance bounds the average deviation of source and target features measured in pixels. In our experiments, we generally set this tolerance to 5% of the fragment radius.

Composition. In order to avoid the accumulation of alias and resampling effects, we compute pixel colors by oversampling and low pass filtering when copying fragments. In the current implementation we use a jittered 9×9 grid of samples. To obtain a seamless composition of the target fragments, we apply a variant of the gradient correction. In contrast to Sun et al. [31], we do not set all gradients on the common boundary to zero but only at those pixels where the gradient in the source fragment is below a certain theshold. By this we preserve strong gradients and avoid or at least reduce color bleeding in image regions with high contrast (see Fig. 7).

Fig. 7a–c. Gradient correction. **a** shows a target fragment after copying the pixels from the source fragment. There are obvious seams in the image. Using gradient correction as proposed by Sun et al. [31] we get the solution in **b** with strong color bleeding. **c** shows our simple improvement maintaining high gradients from the source fragment

Remarks and extensions. In practice, due to the perspective corrections, there is no need to search for fragments with different scalings since this is taken care of by the rectifications. For the same reason rotations also turned out to be of minor relevance in our experiments. In our implementation we use the FFT library *FFTW*[15] which is optimized for image sizes of $2^a\,3^b\,5^c\,7^d\,11^e\,13^f$, $a,b,c,d \geq 0$, $(e+f) \in \{0,1\}$ pixels. This covers a wide range of different image formats. As we demonstrate in Sect. 9, the Fast Fourier Transform is quite efficient such that we can afford a relatively large number of source buffers with high image resolutions and large fragments without compromising the interactivity of our system. Notice that the complexity of the FFT-based fragment search does not depend on the size of the fragments at all.

Due to the extremely fast FFT-based fragment matching procedure, we can actually afford to do several fragment searches when the user clicks on a certain pixel location. In particular, we can check at other pixel locations in the vicinity of the pixel, which the user has selected, if there are better matches. By this we implement an effective pixel snapping technique such that the user has to click less precisely while the system snaps to a nearby pixel which allows for a superior fragment match.

9 Results and discussion

We have tested our system on a large number of examples. All the completions shown in this section have been

generated on an AMD Athlon64 3500+ system. The user interaction for each example took several seconds to several minutes depending on the size and complexity of the task (see the accompanying video [29]).

Trying to complete images with extreme perspective distortion as shown in Fig. 1 without using perspective correction leads to strong perspective artifacts. This problem can be alleviated when using different scales as Drori et al. [12]. When using structure propagation as proposed by Sun et al. [31], perspective artifacts can be avoided to some extend but at the cost of extensive user interaction. However, for the simple cube-example in the last row of Fig. 9, there is no possibility to complete the missing area appropriately with structure propagation only. Our system solves all these completion tasks by using the proper rectification homographies.

In Fig. 9 we demonstrate the flexibility and simplicity of our method on different high-resolution images with difficult completion tasks. The Colosseum (1024×768) and the old arc (1280×882) are examples for using our method to restore old buildings by removing damaged parts and completing the missing areas. Old arc was an especially challenging example because of the high similarity of the colors. In order to improve the matching procedure the user adds an additional structure layer. The Gallery (1024×768) shows how we use our method for perspectively correct continuation of the structure in images. Because our method is not restricted in terms of the number of input buffers we can use multiple images as input. Figure 8 shows a complex example where the completion of a building in the main image is done by using information from an additional image showing the backside of the same building (both 1280×960).

In the accompanying video [29], we show completion results with our system for images taken from Drori et al. [12] and from Sun et al. [31].

Limitations. As any other image completion approach, we cannot complete a part of the image in a meaningful manner if the necessary information is not contained in the known region of the input. However, due to our fast fragment search, we can at least extend our search space to several images without seriously compromising the response time of our system.

Our perspective correction is still a plane-to-plane mapping, which means that varying depth *within* a fragment cannot be handled properly. This can be seen quite clearly in the Colosseum example in Fig. 9 where some arcs have a wrong perspective after the completion. To avoid this, a much more detailed manual annotation of the image would be necessary.

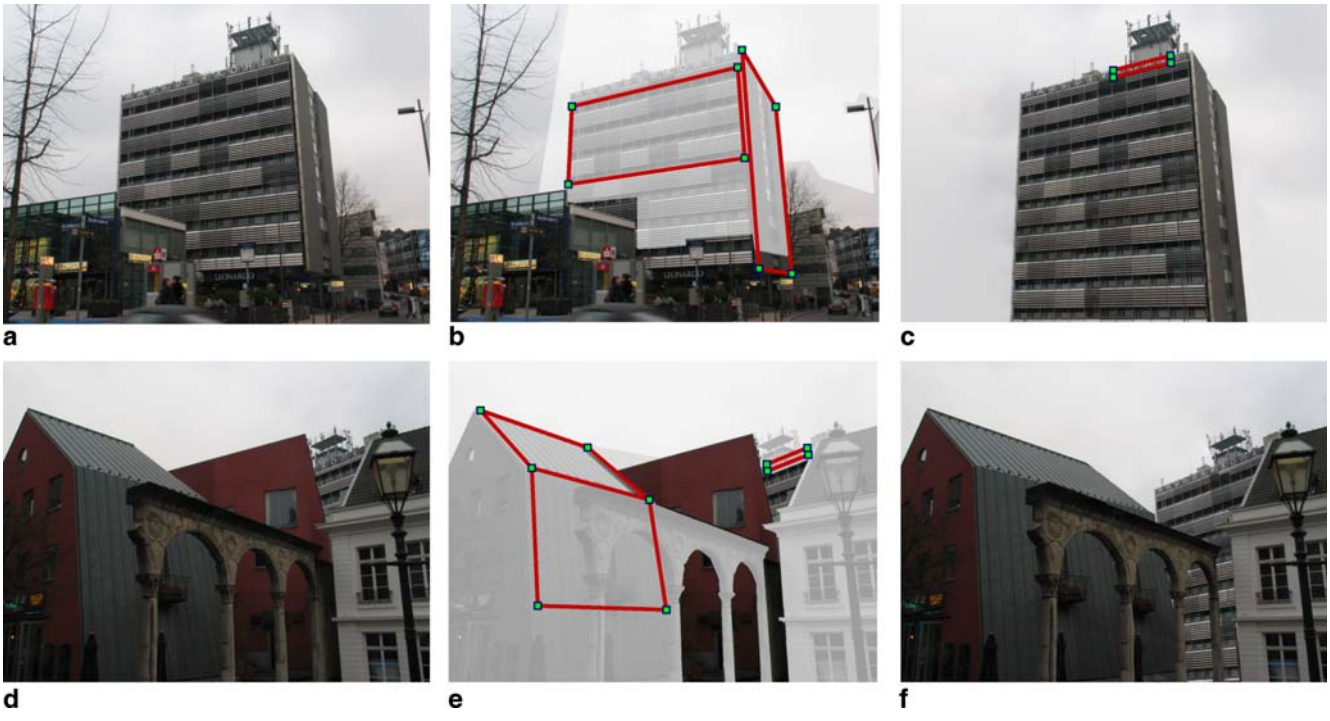

Fig. 8a–f. Multiview. This example shows how our system can combine information from several photos. Images **a** and **d** are the input. First, **a** is completed using the user information depicted in **b** and the result is shown in **c**. Then, we use **c** and **d** with the user interaction shown in **c** and **e** to generate the final solution in **f**. The quads drawn in **c** and **e** specify the rectifications which establish the correspondence between the two input images

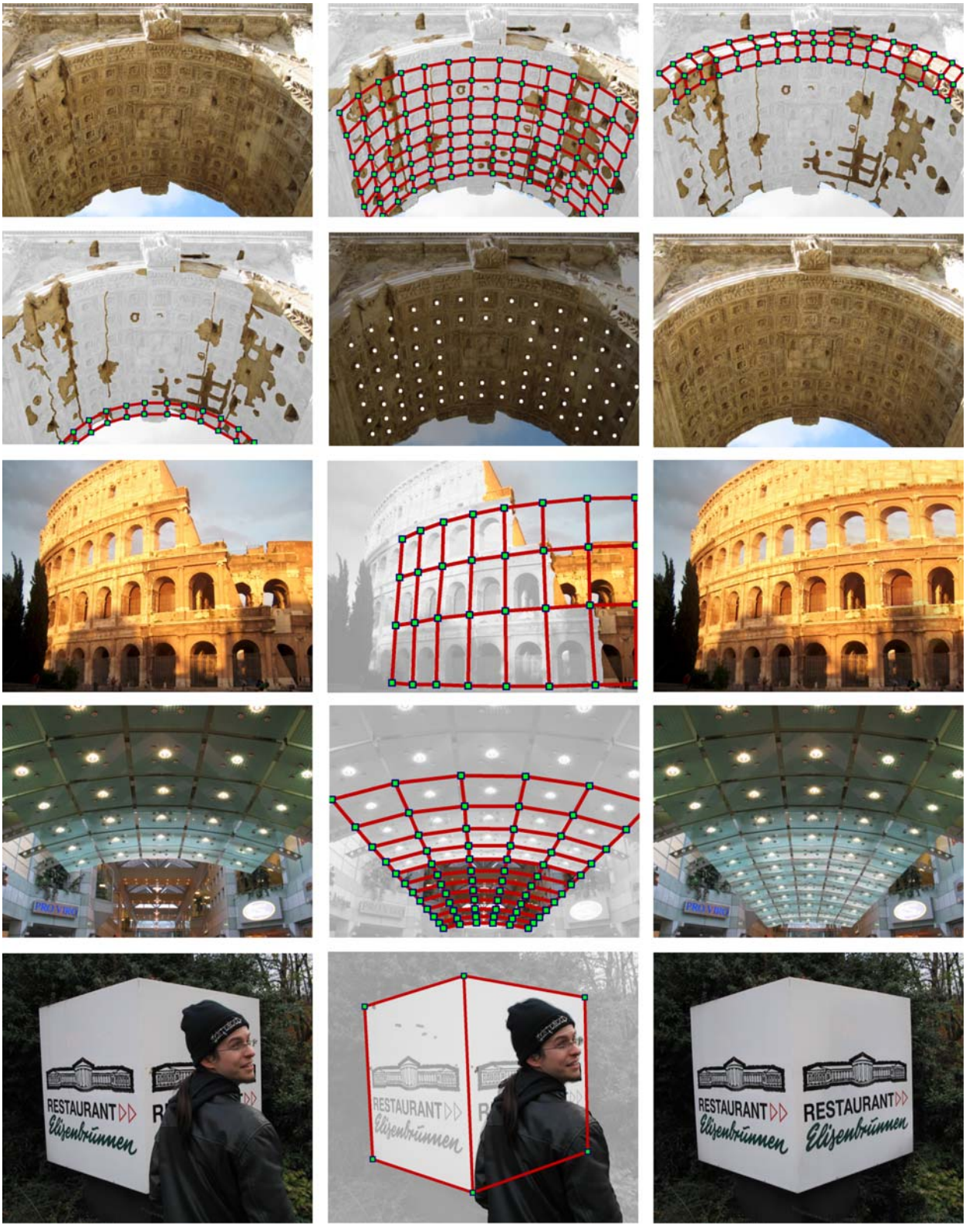

Fig. 9. Results. This figure shows complex completion examples: for the top example (old arc, 1280×882) we used three different grids and one feature layer for the restoration. The feature layer guarantees a proper alignement of the fragments to the structure of the arc. The Colosseum (1024×768), gallery (1024×768) and cube (1280×1120) examples were completed by using only one quad-grid each

10 Conclusion

We have presented a novel system for image completion which allows for perspective corrections of the copied fragments. The correction is based on rectifying homographies, which are interactively defined by the user via simple 2D interaction metaphors. Using these metaphors is easy and intuitive and allows for completing images which previous approaches cannot handle.

By exploiting FFT efficient source fragment search in the Fourier domain can be implemented. We showed that most operations for image completion, including validity checking and feature preservation, can be formulated as a sequence of convolution operations and hence can be computed efficiently.

Future work. We would like to link our image completion system with a large database of images. By storing the images' Fourier transforms, we can quickly search a large number of images for the best matching fragment. Building such a database would require manual preparation to specify the homographies for each image.

Another research direction is to extend our method to the completion of video footage by using a 3D Fourier transform and searching for space-time fragments. A problem here is to find a sufficiently simple user interface to specify and propagate per-frame rectifications.

Acknowledgement Colosseum image is the courtesy of Clint Morgan, Computer Science Department, University of New Mexico. The original rider image used for the Fig. 6 is taken from Sun et al. [31]. We would like to thank Arne Schmitz for his help during the creation of the figures for this paper.

References

1. Arya, S., Mount, D.M.: Approximate nearest neighbor queries in fixed dimensions. In: SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms, pp. 271–280. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1993)
2. Ashikhmin, M.: Synthesizing natural textures. In: SI3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics, pp. 217–226. ACM Press, New York, NY, USA (2001)
3. Ballester, C., Bertalmio, M., Caselles, V., Sapiro, G., Verdera, J.: Filling-in by joint interpolation of vector fields and gray levels. In: IEEE Transactions on Image Processing, vol. 10, pp. 1200–1211 (2001)
4. Barrett, W.A., Cheney, A.S.: Object-based image editing. In: SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, pp. 777–784 (2002)
5. Bertalmio, M., Bertozzi, A., Sapiro, G.: Navier-stokes, fluid dynamics, and image and video inpainting. In: Proc. of Conf. Computer Vision and Pattern Recognition, vol. 1, pp. 355–362 (2001)
6. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 417–424. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
7. Bertalmio, M., Vese, L.A., Sapiro, G., Osher, S.: Simultaneous structure and texture image inpainting. In: CVPR (2), pp. 707–712 (2003)
8. Boehm, W., Prautzsch, H.: Geometric Concepts for Geometric Design. A K Peters, Ltd. (1994)
9. Criminisi, A., Pérez, P., Toyama, K.: Object removal by exemplar-based inpainting. In: CVPR (2), pp. 721–728 (2003)
10. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: SCG '04: Proceedings of the twentieth annual symposium on Computational geometry, pp. 253–262. ACM Press, New York, NY, USA (2004)
11. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In: SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 11–20 (1996)
12. Drori, I., Cohen-Or, D., Yeshurun, H.: Fragment-based image completion. ACM Transactions on Graphics **22**(3), 303–312 (2003)
13. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 341–346. ACM Press, New York, NY, USA (2001)
14. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2, p. 1033 (1999)
15. Frigo, M., Johnson, S.G.: The design and implementation of FFTW3. Proceedings of the IEEE **93**(2), 216–231 (2005). Special issue on Program Generation, Optimization, and Platform Adaptation
16. Harrison, P.: A non-hierarchical procedure for re-synthesis of complex textures. In: International Conference in Central Europe on Computer Graphics and Visualization, pp. 190–197 (2001)
17. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press (2003)
18. Hel-Or, Y., Hel-Or, H.: Real time pattern matching using projection kernels. In: ICCV, pp. 1486–1493 (2003)
19. Horry, Y., Anjyo, K.I., Arai, K.: Tour into the picture: using a spidery mesh interface to make animation from a single image. In: SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, pp. 225–232 (1997)
20. Jia, J., Tang, C.K.: Image repairing: Robust image synthesis by adaptive nd tensor voting. In: CVPR (1), pp. 643–650 (2003)
21. Kasson, J.M., Plouffe, W.: An analysis of selected computer interchange color spaces. ACM Trans. Graph. **11**(4), 373–405 (1992)
22. Komodakis, N., Tziritas, G.: Image completion using global optimization. In: CVPR06, pp. I: 442–452 (2006)
23. Kwatra, V., Essa, I., Bobick, A., Kwatra, N.: Texture optimization for example-based synthesis. ACM Trans. Graph. **24**(3), 795–802 (2005). DOI http://doi.acm.org/10.1145/1073204.1073263
24. Kwatra, V., Schoedl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: Image and video synthesis using graph cuts. ACM Transactions on Graphics, SIGGRAPH 2003 **22**(3), 277–286 (2003)
25. Liang, L., Liu, C., Xu, Y.Q., Guo, B., Shum, H.Y.: Real-time texture synthesis by patch-based sampling. ACM Trans. Graph. **20**(3), 127–150 (2001)
26. Liu, Y., Lin, W.C., Hays, J.: Near-regular texture analysis and manipulation. ACM Trans. Graph. **23**(3), 368–376 (2004)
27. Oh, B.M., Chen, M., Dorsey, J., Durand, F.: Image-based modeling and photo editing. In: SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 433–442. ACM Press, New York, NY, USA (2001)
28. Oliveira, M.M., Bowen, B., McKenna, R., Chang, Y.S.: Fast digital image inpainting. In: Proceedings of the International Conference on Visualization, Imaging and Image Processing, pp. 261–266 (2001)

29. Pavić, D., Schönefeld, V., Kobbelt, L.: Accompanying video. http://www.rwth-graphics.de/downloads/
30. Pérez, P., Gangnet, M., Blake, A.: Patchworks: Example-based region tiling for image editing. Tech. Rep. MSR-TR-2004-04, Microsoft Research (2004)
31. Sun, J., Yuan, L., Jia, J., Shum, H.Y.: Image completion with structure propagation. ACM Trans. Graph. **24**(3), 861–868 (2005). DOI http://doi.acm.org/10.1145/1073204.1073274
32. Wei, L.Y., Levoy, M.: Fast texture synthesis using tree-structured vector quantization. In: SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 479–488. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
33. Wilczkowiak, M., Brostow, G.J., Tordoff, B., Cipolla, R.: Hole filling through photomontage. In: 16th British Machine Vision Conference 2005 - BMVC'2005, Oxford, United Kingdom, pp. 492–501 (2005)

DARKO PAVIĆ received his diploma degree in computer science in 2004 and joined in the same year the Computer Graphics Group, both at the RWTH Aachen University, Germany, where he is currently pursuing his Ph.D. degree. His research interests mainly focus on mesh and image repair, image and video processing.

VOLKER SCHÖNEFELD is an undergraduate computer science student and member of the Computer Graphics Group at the RWTH Aachen University, Germany.

LEIF KOBBELT is a full professor and the head of the Computer Graphics Group at the RWTH Aachen University, Germany. His research interests include all areas of Computer Graphics and Geometry Processing with a focus on multiresolution and free-form modeling as well as the efficient handling of polygonal mesh data. He was a senior researcher at the Max-Planck-Institute for Computer Sciences in Saarbrücken, Germany from 1999 to 2000 and received his Habilitation degree from the University of Erlangen, Germany where he worked from 1996 to 1999. In 1995/96 he spent a post-doctorate year at the University of Wisconsin, Madison. He received his master's degree in (1992) and Ph.D. in (1994) from the University of Karlsruhe, Germany. Over the last few years he has authored many research papers in top journals and conferences and served on several program committees.