Silhouette-Aware Warping for Image-Based Rendering

Gaurav Chaurasia, Olga Sorkine, George Drettakis

Eurographics Symposium on Rendering 2011
Ravi Ramamoorthi and Erik Reinhard
(Guest Editors)

Volume 30 (2011), Number 4

Silhouette-Aware Warping for Image-Based Rendering

Gaurav Chaurasia[1], Olga Sorkine[2],George Drettakis[1]

[1]REVES/INRIA Sophia Antipolis, France
[2]ETH Zurich, Switzerland

Abstract

Image-based rendering (IBR) techniques allow capture and display of 3D environments using photographs. Modern IBR pipelines reconstruct proxy geometry using multi-view stereo, reproject the photographs onto the proxy and blend them to create novel views. The success of these methods depends on accurate 3D proxies, which are difficult to obtain for complex objects such as trees and cars. Large number of input images do not improve reconstruction proportionally; surface extraction is challenging even from dense range scans for scenes containing such objects. Our approach does not depend on dense accurate geometric reconstruction; instead we compensate for sparse 3D information by variational image warping. In particular, we formulate silhouette-aware *warps that preserve salient depth discontinuities. This improves the rendering of difficult foreground objects, even when deviating from view interpolation. We use a semi-automatic step to identify depth discontinuities and extract a sparse set of depth constraints used to guide the warp. Our framework is lightweight and results in good quality IBR for previously challenging environments.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.3]: Picture/Image Generation—Display algorithms;

1. Introduction

Image-based rendering (IBR) [DTM96, LH96, GGSC96] is a powerful approach to easily capture and display 3D environments from photographs. Recent advances in computer vision and geometry processing make it possible to take 10-20 photographs, use automatic camera calibration [SSS06] and multi-view stereo to obtain dense depth/disparity maps [GSC*07, FP09]. A final surface reconstruction step [KBH06] can then be used to merge these depth maps. The resulting approximate 3D geometry or *proxy* can be rendered by re-projecting the input photographs onto the proxy and blending closest views in the style of Unstructured Lumigraph Rendering (ULR) [BBM*01]. This method combines recent vision techniques with ULR to achieve what we believe is the best result possible in our context, using existing solutions. Nonetheless, the solutions described above still have several limitations. First, multi-view stereo and reconstruction approaches have difficulty producing 3D geometry of sufficiently good quality for foreground objects with complex shapes such as trees, or sharp depth discontinuities such as vehicles parked in front of façades. Such situations are very frequent, especially in urban scenes. Consequently, IBR approaches that rely on accurate geometry can suffer from artifacts for such scenes. Second, most recent fully image-based methods which can handle complex scenes [SLW*08, MHM*09], have been developed only for small "baselines" and most often operate on paths interpolating input camera positions and orientations. Lastly, for many cases it is possible to improve the quality of 3D reconstruction by taking more images. However, for complex foreground objects reconstruction often does not improve, while leading to more "heavyweight" capture.

We present a new approach which addresses these limitations. Our central idea is to compensate for incorrect or incomplete geometric information by introducing silhouette-aware variational warping [SS09]. We focus on scenes containing complex foreground geometry that is hard to reconstruct and on *wide baseline* IBR, i.e., with a large camera displacement between photographs.

We first manually select important depth discontinuities, or *silhouettes*, and generate a uniform reconstructed point cloud for each image which guide our *silhouette-aware warp*. We thus preserve important depth discontinuities during the warp operation by letting parts of the mesh around the silhouettes become very "elastic". Finally, we present an efficient multi-pass rendering algorithm which blends the

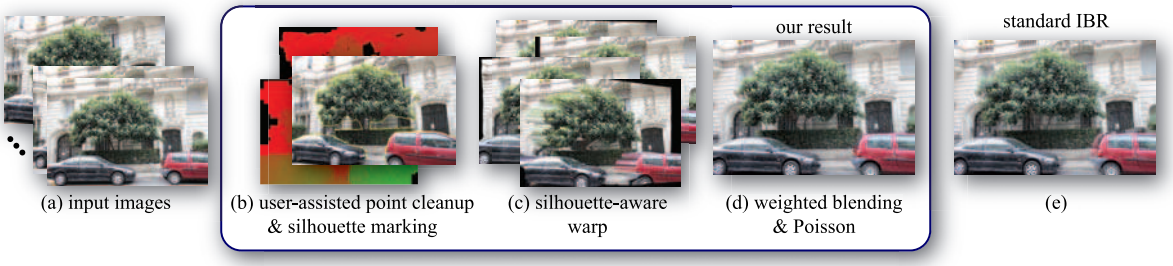


Figure 1: *(a) We use 10-20 input photographs and multi-view stereo to create a dense 3D point cloud. (b) With our user-assisted point pre-processing the user designates important silhouettes and we reduce the cloud to ∼6,000 points per image. (c) The silhouettes and 3D point cloud guide a* silhouette-aware warp, *applied to 4 images at each frame. (d) Our renderer generates a high-quality final image which handles hard cases such as trees and other foreground objects. (e) Compared to previous geometry-based methods, many artifacts are removed.*

warped images. We define appropriate weights for elastic vs. non-elastic parts of blended images, and post-process to provide a trade-off between blurring and color discontinuities.

Previous approaches [DRE*10, DCC*09] directly reproject warp mesh vertices into a novel view, resulting in a "hard warp". Such approaches require dense depth maps [DRE*10] or warp mesh vertices tracked over time [DCC*09]. Since our setting typically has sparse or unreliable depth information, temporal jumping artifacts ensue [LGJA09]. To avoid these artifacts, we use a variational warp with sparse depth information as guiding constraints. Mesh cutting is used to handle occlusion in hard warps [DCC*09]. Such a solution is however inapplicable with smooth variational warping, which is why we introduce a *silhouette-aware* formulation (see Sect. 2 for further discussion).

Our main contributions can be thus summarized as follows:

- The representation consisting of sparse depth constraints and silhouette edges, which enables depth-preserving variational warping for wide-baseline IBR.
- The introduction of *silhouette-aware* warping for IBR in which "elastic" edges absorb distortions while depth discontinuities are preserved.
- An efficient rendering algorithm with a good trade-off between blurring and color discontinuities.

Our approach greatly reduces artifacts compared to the best combination of state-of-the art reconstruction and IBR techniques, while overcoming the limitations discussed above (see results Fig. 7). In particular we treat scenes with hard-to-reconstruct objects and viewing paths which do not interpolate the input cameras. Finally, only a small number of images is required, resulting in a lightweight capture process.

2. Previous Work

The earliest work in image-based rendering (IBR) did not use any geometric information about the scene [MB95, LH96] and depended on a large number of input images to avoid rendering artifacts. Other approaches such as the Lumigraph [GGSC96] and view-dependent texture mapping [DTM96] used an approximate geometric *proxy* to create novel views with fewer input images. Unstructured Lumigraph (ULR) [BBM*01] generalized this to arbitrary unstructured input cameras; this method remains the most general free-viewpoint IBR algorithm.

Modern graphics hardware can be used to enhance ULR by per-pixel blending and visibility handling [EDM*08]. The results improve dramatically with better geometric reconstruction. Consequently, there has been a lot of recent work coupling 3D reconstruction and IBR, mainly focusing on developing more accurate geometric proxies.

Multi-view stereo. A detailed survey of multi-view stereo algorithms is described in [SCD*06]. The development of structure-from-motion for automatic camera calibration [SSS06] has further accelerated multi-view stereo research. Goesele et al. [GSC*07] extract dense depth/disparity maps from input images which then can be merged using surface reconstruction techniques, e.g., Poisson reconstruction [KBH06]. Patch-based multi-view stereo [FP09] is one of the most general multi-view stereo algorithms which extracts 3D point clouds for all classes of scenes: outdoor, objects, indoors, etc. Hornung and Kobbelt [HK09] propose a generalized approach for stereo reconstruction and rendering on the GPU, avoiding the need for an explicit proxy. However, their method effectively builds on standard stereo reconstruction; complex scenes with wide baselines thus become hard to handle.

Another class of algorithms has been developed specifically for IBR applications. Piecewise-planar reconstruc-

tion [SSS09] and Manhattan-world priors [FCSS09] have been used for improved IBR. These approaches bypass surface reconstruction and produce lightweight proxies consisting of planes. However, they often have difficulties for scenes with non-planar or irregular geometry, or complex foreground objects, such as vehicles or trees.

Image interpolation. Image interpolation has been used to transform two neighboring images of the same scene separated in space and/or time to generate in-between images without any 3D scene information. Chen and Williams [CW93] interpolated dense optical flow between input images, Seitz and Dyer [SD96] generated target views on the line joining the optical centers of two input views assuming no occlusions, and Lhuillier and Quan [LQ99] interpolated matched image regions. More recently, perceptually-motivated image interpolation was described in [SLW*08] and spatio-temporal view interpolation was presented in [LLB*10]. A high-quality approach for interpolating two images is presented in [MHM*09]; they preserve frequency content of images by using graph-cut to create seamless transitions.

These approaches are powerful and robust but require a very small baseline between images. Mahajan et al. [MHM*09] report a 30-pixel maximum baseline, with 2D correspondences becoming unreliable beyond this limit. Moreover, they are restricted to view interpolation paths.

IBR has been used in a different context in [SSS06] for navigating photo collections. This is further generalized to video navigation in [BBPP10] which uses scene geometry and foreground segmentation to reduce artifacts on the salient scene object (e.g., performer) during view interpolation. Reconstructed point clouds are used in [GAF*10] for wide baseline view interpolation of prominent scene object. Their goal is to generate visually pleasing transitions in an image sequence rather than photo-realistic novel views of the whole scene.

Image warping. Modifying the image content by means of warping has been used for morphing, image editing, artistic manipulation [CAA10] and re-targeting [SS09]. In variational warping, an objective functional is formulated according to the image content and the desired goal. Such goals can be preservation of salient features while resizing [GSCO06] or attaining an optimal disparity for a stereo image pair [LHW*10]). This functional is then optimized on a grid mesh overlaid on the image subject to boundary constraints. So far, mostly 2D constraints (e.g., new boundary dimensions) have been employed; in contrast, we use reconstructed 3D points to guide the global warp.

A related approach that uses reprojected 3D constraints is the video stabilization technique of [LGJA09], which warps frames of an input video to a novel viewpoint that lies on a smoothed camera path. However, their approach requires small baseline warping without (dis)occlusions. Our technique includes a novel *silhouette-aware warp* formulation to handle these cases which are very common for wide-baseline warping. Further differences are discussed in Sec. 5.2.

When dense, per-pixel depth [DRE*10] is available or mesh vertices are tracked across successive frames [DCC*09] direct re-projection into the novel view can be used. This gives a "hard warp" (see also Sect. 1). However, in our wide baseline multi-view setting, mesh vertices may not be reliably reconstructed in neighboring views. and thus hard warps result in jumping artifacts as observed in [LGJA09]. To provide smoother temporal effects, we use variational warping.

In this variational warping context, we need to incorporate silhouettes and "elastic edges" to absorb distortion, to preserve complex foreground object shapes. The mesh cutting approach of [DCC*09] would not work well with a variational approach. Specifically, it is impractical to set up a separate variational warp for each depth layer of scene since silhouettes are generally not closed and multiple objects would give too many separate layers for each image. In addition, special treatment, such as inpainting or texture synthesis would be necessary for all "cut" regions.

3. Overview

The input to our method is a set of images calibrated using Bundler [SSS06] and a dense 3D point cloud generated using [FP09]. Our approach has three main steps (see Fig. 1):

Pre-processing. Our approach first selects silhouettes around complex foreground objects (trees, cars, etc.) for each input image (Sec. 4.1). These silhouettes are used to correctly handle depth discontinuities. A 3D point cleanup step then decimates the dense multi-view stereo point cloud down to a sparse point cloud with uniform spatial distribution. This step also fills in poorly reconstructed regions using depth from neighboring points (Sec. 4.2). The resulting sparse point set provides stable constraints for our image warp.

Silhouette-aware image warp. The points from each input image are mapped to their respective desired final positions by reprojecting them into the novel view. These act as guiding constraints for our image warp in the form of projection energy. A similarity transform energy prevents deformation of warp mesh triangles. We define "elastic" triangles around silhouettes which absorb the distortion because of depth differences. The last energy term minimizes warping artifacts that distort the shape of silhouettes (Sec. 5.2).

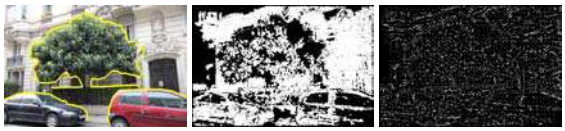
Rendering. At any pixel we blend two images to get correct motion parallax and fill in disoccluded regions. We define appropriate weights to correctly diminish the visual impact of strong distortion produced by the elastic edges around the silhouettes. Finally, we use an optional Poisson synthesis step to alleviate seams.

Our image warping works for poorly reconstructed ob-

jects because the silhouettes segment the image into contiguous regions at different depths. The sparse but uniform reconstruction given by 3D point processing provides enough constraints for correct 2D warping of each region, resulting in significant quality improvement compared to methods which rely on accurate geometry.

4. Extracting the Silhouette and Sparse Constraint Representation

Our approach requires pre-annotated silhouettes (Fig. 2(a)) and uniform distribution of reconstructed points on each image (Fig. 2(a),(c)), both of which can be provided by a variety of approaches. Our core image-based rendering approach is independent of the methodology used for providing either of these.



(a) (b) (c)

Figure 2: *(a) Silhouettes marked at sharp depth differences; (b) Original reconstructed point cloud with 118,049 points; (c) Sparse uniform point cloud with 6324 points selected by our approach. Note the regions with no original 3D points are also filled with new points (see Sec. 4.2).*

4.1. Silhouette Selection

Silhouettes can be manually authored in each input image or computed (semi) automatically. Modern image segmentation algorithms [AMFM11] can be used to extract image boundaries automatically. Considering the importance of silhouettes for a variety of applications (object recognition, 3D reconstruction etc.), segmentation techniques have been adapted for extracting silhouettes or occlusion boundaries from a single image [HSEH07] or motion sequences [SH09, HY10]. Even though the edge maps returned by these approaches are impressive, they often have many false positives which need to be removed manually and missing edges have to be manually added. In addition, edge-maps have to converted to binary maps using a dataset dependent threshold. Then, they need to be converted into polygonal curves using chaining [TC89] and line segment decomposition by Douglas-Peucker algorithm [DP72]. Noisy edge-maps such as those in our scenes can make polygonal approximation ambiguous.

We have experimented with such approaches extensively, both by applying them directly, and developing direct extensions. We observed that, in practice, segmentation followed by same degree of user interaction does not give the same accuracy as manual authoring and can actually be longer than direct manual edge marking. Please refer to supplementary material for a summary of our extensive tests and a comparison of automatic methods with manual authoring.

In view of the above, we prefer manual silhouette authoring over a combination of segmentation and user intervention. Manual authoring took 40-60 seconds for each image in our datasets (see video). This is much faster compared to the time needed to manually create pixel-accurate geometry by hand. For objects such as trees, such geometry creation would require a skilled and experienced modeler and even then would probably require hours of work.

We expect that the silhouette extraction step can be largely automated with advances in the state of the art (see discussion in Sec. 8).

4.2. 3D Point Selection

The goal of this step is to retain a uniform distribution of 3D points over the image, fill regions that have few or no points and remove erroneous points near silhouettes or specular regions.

We select a uniform spatial distribution by splatting the points on the image with a large splat size (21×21 or more) and select the suitably-sized subset of points which cover the largest number of pixels. This approach gives regularly spaced points. Poorly reconstructed regions such as holes in the point set are filled by adding new points placed at the depth of nearest available point.

To avoid mixing foreground and background points on either side of the silhouettes, our approach conservatively removes all existing points within a small distance of silhouettes and replaces them with points using the depth from their respective side. This ensures that the silhouettes clearly separate points with different depths. We observed that such narrow regions are too small to contain significant depth gradients. Hence, this does not compromise warp accuracy. It is important to note that the newly added points are generated on a per-image basis and have no correspondence across images. As such, they are not true 3D points and do not augment the reconstruction. They simply provide constraints for stabilizing the image warp described next.

Some regions such as specular surfaces have incorrect 3D points, which are removed manually. This step is required only if a large number of incorrect 3D points are observed in a certain region.

In our examples, the reconstruction produced 120K-200K points. Using the process described above, we retained 5-6K points for each image, which we call the set $\mathcal{P}_i$. We observed that 6-9K points did not improve the warp quality and less than 3K points led to warping artifacts. The optimal number of points depends on desired output image resolution. A higher desired level-of-detail would require more constraints

for the warp, hence more points.The entire process, including user interaction (if needed), took about 4 minutes for a dataset of 15 images. For more implementation details, please refer to supplementary material.

5. Image Warping using 3D Constraints

Given a novel view, expressed by a camera projection matrix C_n, our goal is to warp the input images $I_1, I_2, \ldots, I_N$ so that they match the actual scene as it would have appeared in that view as faithfully as possible. We then use the warped images in the IBR pipeline (Sec. 6).

Denote by C_i the camera projection matrix of input image I_i. If we knew the mapping U_i from every pixel $\mathbf{q} \in I_i$ to the corresponding 3D point $\mathbf{p} \in \mathbb{R}^3$ that is captured in this pixel ($C_i(\mathbf{p}) = C_i(U_i(\mathbf{q})) = \mathbf{q}$), then the warp of image I_i into the new view would be simply

$$W = C_n \circ U_i. \tag{1}$$

However, we do not have a dense per-pixel 3D reconstruction of the scene. On the contrary, we wish to use only a small set of 3D points that suffices for effective image warping. We therefore replace the per-pixel warp above with a *sparse* set of constraints on pixel positions and a warp prior that dictates the warping function to be smooth (except where (dis)occlusions happen) and locally preserve the shape of the image content. We handle occlusions by explicitly modeling the desired warp behavior along silhouettes, as will be described in Sec. 5.2. Together, the positional constraints and warp behavior priors define an energy functional E, and we find the warp that minimizes this energy using variational optimization.

Setup. In order to compute the optimized variational warp, we discretize the image domain by overlaying a triangle mesh $\mathcal{M}_i$ on image I_i, with vertices $\mathcal{V}_i = \{\mathbf{v}_1, \ldots, \mathbf{v}_{N_i}\}$ and faces $\mathcal{F}_i$. We denote the warped vertex positions as $\mathbf{v}'_j = W(\mathbf{v}_j)$. The warp function W minimizes the energy functional $E(W)$ which we describe next. W is piecewise-linear (linear within each mesh face); therefore, to compute it we need to find the warped positions $\mathbf{v}'_j$.

5.1. Smooth Image Warping

3D constraints. Let $\mathcal{P}_i \subseteq \mathbb{R}^3$ be the set of 3D points whose projected location in image I_i is known, obtained with the method of Sec. 4. For each point $\mathbf{p} \in \mathcal{P}_i$ we have a corresponding 2D position $\mathbf{q}$ in the image, i.e., $\mathbf{q} = C_i(\mathbf{p})$. The 3D points should be projected correctly onto the novel view C_n; therefore the warp should satisfy

$$W(\mathbf{q}) = C_n(\mathbf{p}). \tag{2}$$

To formulate the above constraint in terms of the mesh vertices, denote the triangle that $\mathbf{q}$ belongs to by $(j,k,l) \in \mathcal{F}_i$, and let $\alpha(\mathbf{q}), \beta(\mathbf{q}), \gamma(\mathbf{q})$ be the barycentric coordinates of $\mathbf{q}$ w.r.t. that triangle. The least-squares energy term for the 3D warp constraints is thus

$$E_p(W) = \sum_{\mathbf{p} \in \mathcal{P}_i} \left\| \left(\alpha(\mathbf{q})\mathbf{v}'_j + \beta(\mathbf{q})\mathbf{v}'_k + \gamma(\mathbf{q})\mathbf{v}'_l\right) - C_n(\mathbf{p}) \right\|^2. \tag{3}$$

We could have imposed Eq. (2) as a hard constraint by including the points $\mathbf{q}$ as vertices of the triangulation $\mathcal{M}_i$; however, as pointed out in [LGJA09], this can lead to temporal incoherence.

Similarity warp prior. To minimize the distortion caused by the warp, we would like W to be locally shape-preserving. We therefore use a similarity energy term, such that the transformation of each mesh triangle by W is as close as possible to a similarity transformation. Analogous energy terms were used in [LGJA09, ZCHM09, WLSL10].

Consider a mesh triangle $t = (\mathbf{v}_j, \mathbf{v}_k, \mathbf{v}_l)$ and attach a local orthogonal frame to it: $\{\mathbf{v}_k - \mathbf{v}_j,\ R_{90}(\mathbf{v}_k - \mathbf{v}_j)\}$, where R_{90} is a counterclockwise rotation by 90 degrees. Assume that $\mathbf{v}_j$ is the origin of the local frame; $\mathbf{v}_k$ can then be expressed simply as $(1,0)$ in the local coordinate system, and $\mathbf{v}_l$ as some (a,b). If the triangle undergoes a similarity transformation, the local frame remains orthogonal and the coordinates of the triangle's vertices remain the same. The similarity energy term can thus be expressed as

$$E_s(W) = \sum_{t \in \mathcal{F}_i} \left\| \mathbf{v}'_l - \left(\mathbf{v}'_j + a(\mathbf{v}'_k - \mathbf{v}'_j) + b(R_{90}(\mathbf{v}'_k - \mathbf{v}'_j))\right) \right\|^2, \tag{4}$$

where

$$a = (\mathbf{v}_l - \mathbf{v}_j)^T(\mathbf{v}_k - \mathbf{v}_j)/\|\mathbf{v}_k - \mathbf{v}_j\| \tag{5}$$

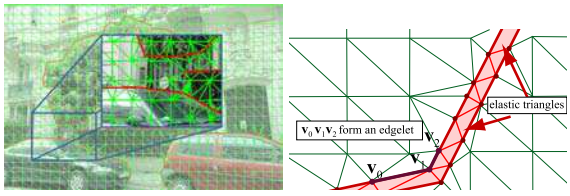
$$b = (\mathbf{v}_l - \mathbf{v}_j)^T R_{90}(\mathbf{v}_k - \mathbf{v}_j)/\|\mathbf{v}_k - \mathbf{v}_j\| \tag{6}$$

are computed from the original mesh.

5.2. Silhouette-aware Warp

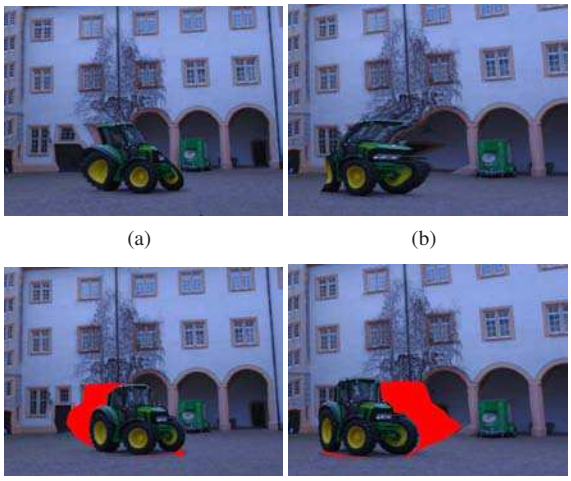
The energies E_p and E_s described so far are minimized by warp functions that are smooth and shape-preserving everywhere. However, we know that the warp should have discontinuities in the vicinity of object silhouettes because of the depth discontinuity there. When considering a small neighborhood around a silhouette edge, the warp may have a discontinuity perpendicular to the edge (to mimic (dis)occlusion) while remaining shape-preserving in the tangent direction. We model this behavior by conceptually inserting a narrow and highly elastic band parallel to the silhouette that is allowed to absorb heavy distortion due to discontinuity (see Fig. 3(b)). The shape of the silhouette itself, on the other hand, is preserved by adding a curve-similarity energy term described below, thus avoiding distortion of foreground objects.

To properly discretize the image domain and formulate the silhouette-specific energy, we take all the silhouette polylines (obtained in Sec. 4) and duplicate them, offsetting the resulting parallel edges by 2 pixels (see Fig. 3(b)). The rest



(a) (b)

Figure 3: *(a) Conforming triangulation used as warp mesh with constrained silhouette polylines shown in red. (b) Two parallel constrained polylines (shown in red) added for the silhouette polyline. Any three consecutive vertices on either of these polylines form an edgelet. The triangles wedged between these edges form the elastic band.*



(a) (b) (c) (d)

Figure 4: *Top row: An input image warped to different novel views without any silhouette handling. Bottom row: Same image warped to same views using our silhouette-aware discontinuous warp. The elastic band that absorbs all the distortion is shown in red.*

of the image domain is sampled uniformly, and we create a constrained conformal triangulation where the (doubled) silhouette edges are fixed (see Fig. 3(a)). All triangles between silhouette lines belong to the *elastic band* and are excluded from the energy term E_s (Eq. (4)), thus allowing the band to be elastic.

To preserve the shape of the silhouette itself, we require the silhouette curve to locally undergo a shape-preserving (i.e., similarity) transformation. The energy formulation is similar to E_s, but it is defined on the silhouette curve this time, instead of a 2D domain. Consider three consecutive vertices lying on the curve, indexed w.l.o.g. as $e = (\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2)$. We call such a sequence of two curve edges an *edgelet* (see Fig. 3(b)). A similarity transformation of the edgelet e means that the angle θ between the two edges, as well as the length ratio $\|\mathbf{v}_0 - \mathbf{v}_1\|/\|\mathbf{v}_2 - \mathbf{v}_1\|$, remains the same. We can therefore write the curve similarity energy term as

$$E_b(W) = \sum_{e \in \mathcal{E}} \left\| (\mathbf{v}'_0 - \mathbf{v}'_1) - \frac{\|\mathbf{v}_0 - \mathbf{v}_1\|}{\|\mathbf{v}_2 - \mathbf{v}_1\|} R_\theta (\mathbf{v}'_2 - \mathbf{v}'_1) \right\|^2, \quad (7)$$

where $\mathcal{E}$ is the set of all edgelets and R_θ is the 2×2 rotation matrix that rotates the edge $(\mathbf{v}_2 - \mathbf{v}_1)$ onto $(\mathbf{v}_0 - \mathbf{v}_1)$. The effect of the above silhouette-aware discontinuous image warp is shown in Fig. 4. The smooth warp described in Sec. 5.1 will cause heavy distortion near depth discontinuities (see Fig. 4(a),(b)).

Liu et al. [LGJA09] use energies E_p and E_s alone and would allow homogeneous distribution of the heavy distortion over the entire image. In contrast, our silhouette-specific energy term E_b preserves the local shape of the silhouettes by absorbing all the distortion in the *elastic* band. When pixels become occluded, the elastic band enables accurate mesh fold-over along the silhouette. When pixels are disoccluded, the elastic band stretches without deforming the silhouette (shown in red in Fig. 4(c),(d)). These bands are later filled using texture from a different image in the final result (explained in Sec. 6). Thus, our image warp is robust to wide-baseline (dis)occlusion.

5.3. Total Warp Energy

The optimal warp W minimizes the weighted sum of the 3D constraints, similarity, and silhouette energies (Eqs. (2), (4), (7)):

$$E(W) = w_p E_p + w_s E_s + w_b E_b. \quad (8)$$

We use $w_s = 1, w_p = w_b = 2$ in our implementation.

The energy $E(W)$ is quadratic in the unknown warped vertex positions $\mathbf{v}'$, therefore it has a unique minimum that is found by solving the sparse linear equation $\nabla E(W) = 0$. We use the direct sparse Cholesky solver TAUCS [Tol03]. Note that the system matrix does not change for different desired views C_n since only the right-hand side of the linear system changes. We therefore precompute the matrix factorization and only perform back-substitutions at runtime, which is very efficient.

Warp optimization implementation. In our experiments, we found initial sampling on a 30×30 vertex grid to be sufficient for 800×600 pixel output frame resolution. This sampling is locally refined to insert the silhouette edges, as described above; we compute the final constrained conformal triangulation using CGAL [Rin10]. Every input image I_i has its own mesh and associated linear system; we thus employ OpenMP to warp multiple images on parallel cores. Finally, the warped meshes are rendered using texture coordinates as their original positions to create the warped images.

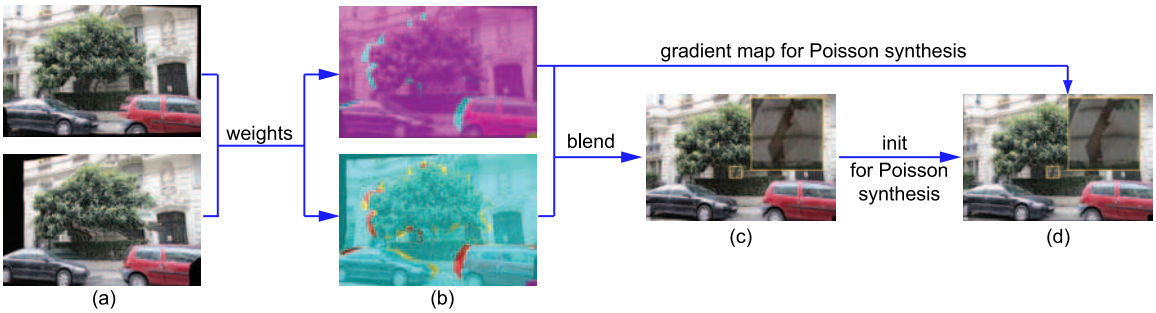


Figure 5: *Rendering pipeline. (a) Warped images. (b) Composite textures $\mathcal{R}_0$ and $\mathcal{R}_1$. Pixels from same warped image are shown in same color. (c) $\mathcal{R}$ generated by blending from $\mathcal{R}_0$ and $\mathcal{R}_1$. (d) Poisson synthesis output $\mathcal{R}'$ using $\mathcal{R}$ as initialization and gradient $\mathcal{G}$ from $\mathcal{R}_0$.*

6. Implementation and Rendering

Our rendering approach has several similarities to ULR [BBM*01]. For clarity we thus summarize this method here.

ULR with per-pixel blending and visibility. A set of images with calibrated cameras and a geometric proxy are taken as input. For every pixel of a novel view, the corresponding 3D point $\mathbf{p}$ on the proxy is found. This point is used to select the input images used, and it is reprojected back into these images. The colors of the corresponding pixels are then blended together using weights that depend on the angle between the view vectors of the input and target views (Fig. 6(a)), as well as the distance to the proxy (see [BBM*01] for exact details).

In our approach the novel view is generated by first selecting a set $\mathcal{W}$ of 4 images which can be used for all pixels in the final image. We observed that 3-4 input images are sufficient to synthesize a novel view. We then warp these images to the target view using our warp formulation in Sec. 5. Recall that the 3D points in $\mathcal{P}_i$ of each image I_i project to the same position in the novel view. As a result, the warped images are well registered.

Blending weights. For each pixel, we select the best two images for blending using a penalty scheme inspired by ULR [BBM*01]. Consider pixel (x,y) of the novel view with center of projection $\mathbf{c}_n$ and a warped input image $I'_i \in \mathcal{W}$ whose center of projection is $\mathbf{c}_i$. Let $\mathbf{p}_n(x,y)$ be the point where the ray shot from $\mathbf{c}_n$ through pixel (x,y) intersects the scene geometry. The required geometry is generated by splatting all 3D points $\cup\mathcal{P}_i$ into the novel view; holes are avoided by extracting depth from a triangulation of $\mathcal{P}_i$. Note that this geometry is typically much coarser than a geometric proxy generated by 3D reconstruction [FP09] and surface extraction [KBH06]. However, our approach is robust to geometric inaccuracies because geometry is only used to compute blending weights.

We use the angle between $(\mathbf{c}_n - \mathbf{p}_n(x,y))$ and $(\mathbf{c}_i - \mathbf{p}_n(x,y))$ as an angle penalty similarly to ULR (see Fig. 6(a)). We also define a "field of view" penalty that checks whether the pixel lies inside the warp mesh $\mathcal{M}_i$ of I'_i (shown in blue in Fig. 6(b)). Our last term penalizes elastic band pixels because the texture in such regions is expected to be heavily distorted (shown in red in Fig. 6(b)).

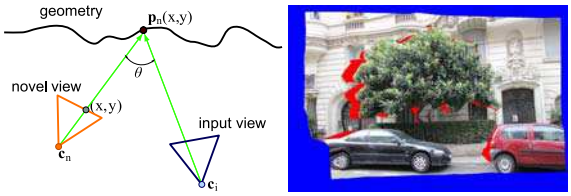


Figure 6: *(a) Diagram showing the angle θ used for $P_{\text{ang}}(I'_i,x,y)$. (b) Warped image showing pixels outside the warp mesh $\mathcal{M}_i$ in blue and pixels inside elastic band in red.*

$$P_{\text{ang}}(I'_i,x,y) = \arccos\left(\langle \mathbf{c}_n - \mathbf{p}_n(x,y),\ \mathbf{c}_i - \mathbf{p}_n(x,y)\rangle\right)$$

$$P_{\text{fov}}(I'_i,x,y) = \begin{cases} \infty & \text{if } (x,y) \text{ lies outside } \mathcal{M}_i, \\ 0 & \text{otherwise} \end{cases}$$

$$P_e(I'_i,x,y) = \begin{cases} \infty & \text{if } (x,y) \text{ lies inside elastic band,} \\ 0 & \text{otherwise} \end{cases}$$

The final penalty is

$$P(I'_i,x,y) = P_{\text{ang}}(I'_i,x,y) + P_{\text{fov}}(I'_i,x,y) + P_e(I'_i,x,y). \quad (9)$$

We create two textures $\mathcal{R}_0$ and $\mathcal{R}_1$, where $\mathcal{R}_0$ is composed of pixels having the lowest penalties from all warped images $I'_i \in \mathcal{W}$ and $\mathcal{R}_1$ the second lowest. A *patch* is a contiguous block of pixels looked up from the same warped image. The patches constituting $\mathcal{R}_0$ and $\mathcal{R}_1$ are shown in different colors in Fig. 5.

Blending weights are calculated from penalties and stored in the alpha channels of $\mathcal{R}_0$ and $\mathcal{R}_1$, respectively. The weights correspond to those used in the original ULR ap-

proach, with the addition of a factor $0 < \Psi \leq 1$:

$$w_{\mathcal{R}_0}(x,y) = 1 - \Psi \frac{P(\mathcal{R}_0,x,y)}{P(\mathcal{R}_1,x,y)},$$
$$w_{\mathcal{R}_1}(x,y) = 1 - \Psi \frac{P(\mathcal{R}_1,x,y)}{P(\mathcal{R}_1,x,y)} = 1 - \Psi. \quad (10)$$

The textures $\mathcal{R}_0$ and $\mathcal{R}_1$ are alpha-blended to give the texture $\mathcal{R}$. The factor Ψ amplifies the difference in the penalties of $\mathcal{R}_0$ and $\mathcal{R}_1$. For example, $\Psi = 1$ would cause $w_{\mathcal{R}_1}$ to always remain 0. If the ratio of penalties $P(\mathcal{R}_0)$:$P(\mathcal{R}_1)$ is 1:2, setting $\Psi = 0.87$ would cause the ratio of weights $w_{\mathcal{R}_0}$:$w_{\mathcal{R}_1}$ to become 4.33:1. This greatly reduces the contribution of $\mathcal{R}_1$ everywhere except where its penalty is very close to that of $\mathcal{R}_0$. In all examples presented here, we have used $\Psi = 0.87$, which gives good results in all our examples.

Poisson synthesis. The blended texture $\mathcal{R}$ may have spatial discontinuities: the patch boundaries or seams of $\mathcal{R}_0$ and $\mathcal{R}_1$ may remain visible, especially the elastic band regions where the texture originates from an image with substantially different view. To create a better final output image, we use Poisson synthesis to essentially inpaint the seam areas more gracefully. We create the gradient map $\mathcal{G}$ from $\mathcal{R}_0$, and its divergence map div$\mathcal{G}$. Note that the gradient inside any patch of $\mathcal{R}_0$ is the same as the gradient of the original image, which is important to retain crisp detail. We force the gradient of the final image $\mathcal{R}'$ to 0 for all pixels lying on patch boundaries $\mathcal{B}_0$ of $\mathcal{R}_0$, which amounts to smooth completion of those areas. Thus, we solve the following Poisson equation:

$$\nabla^2 \mathcal{R}' = \text{div}\mathcal{G} \quad \text{subject to} \quad \nabla\mathcal{R}'\big|_{\mathcal{B}_0} = 0. \quad (11)$$

We do not pose explicit Dirichlet conditions but rather initialize the solver with the blended image $\mathcal{R}$ and perform a few Jacobi iterations. This suffices, since the initial guess is very close to the solution. This (optional) Poisson synthesis step alleviates spatial artifacts, while initialization with $\mathcal{R}$ helps temporal coherence. This can be seen in the inset of Fig. 5 (right). Our approach thus prefers smooth spatio-temporal transitions over blending more images, which leads to ghosting [MHM*09].

7. Results

To provide fair comparisons, we have combined state of the proxy reconstruction with the best set of techniques available for free viewpoint, wide-baseline IBR. In particular, we extend ULR to use per-pixel blending on the GPU, in contrast to vertex blending used in the original method [BBM*01]. We have also added a recent visibility checking algorithm [EDM*08] to further improve quality. In what follows, we call this method IULR for "Improved" ULR.

Given the lack of accurate geometry for foreground objects, IULR results have ghosting artifacts and incorrect occlusion handling. Visibility checking does not alleviate occlusion artifacts because the proxy used for creating visibility maps is erroneous.

Dataset	Best proxy	Same-size proxy	Our approach
Castle-P30	49.1 MB	4.7 MB	2.5 MB
Street-10	25.3 MB	3.6 MB	1.5 MB
Tree-18	21.2 MB	3.0 MB	3.1 MB
Aquarium-20	32.9 MB	15.4 MB	2.9 MB
Yellowhouse-12	26.0 MB	20.0 MB	1.8 MB

Table 1: *Storage for the proxy (used by IULR) and our approach. Proxy sizes are meshes (vertices/faces, no normals/colors/texture coordinates) in ASCII ".obj" format.*

Storage. Our method requires the storage of 5-6k points per image. In contrast, detailed proxies can be quite large for complex scenes with trees etc. (see Table 1). In our analysis, we compare to a "best proxy" which is the geometry of highest resolution and quality that could be extracted from the dataset, and to "same-size proxy" i.e., a proxy of similar size to our storage requirements,

We have tested our approach on challenging datasets which cannot be reconstructed accurately. *Castle-P30* is a standard multi-view stereo dataset [SvHG*08] with a foreground object (tractor) in very wide baseline images. Piecewise-planar reconstruction [SSS09] gives unacceptable artifacts on the tractor. *Aquarium-20* has multiple foreground objects at different depths, which are known to be difficult to handle [MHM*09]. *Street-10* and *Tree-18* show our results for general urban scenes with vehicles and trees. The *Tree-18* dataset had many incorrectly reconstructed points on the tree, which were manually removed. The presence of vegetation makes 3D reconstruction and surface extraction very difficult; manually modeling such scenes is also very difficult and tedious. The baselines in our datasets vary from 275 pixels (14% of image height) in *Aquarium-20*; 260 pixels (16%) in Street-10 to 530 pixels (24%) in *Castle-P30*. With minimal user input, our approach generates much improved quality novel views from a free-viewpoint camera path even for poorly reconstructed datasets.

Performance. We tested our method on an Intel Xeon (2.8 Ghz) running Windows 7 with NVIDIA Quadro 6000. Setting up the warp mesh and factoring the linear system for each input image with TAUCS takes 2-13 seconds. At run time, warping 4 input images on parallel cores takes 8-22 ms. The overall frame rate is 15 FPS with Poisson synthesis and 30-35 FPS for 1600×1200 size render targets.

8. Discussion and Future Work

The results show that overall, our approach has significantly fewer disocclusion and ghosting artifacts compared to Improved ULR, even when the "best proxy" is used. With "same-size proxy" the improvement is more pronounced. This indicates two advantages of our method: (i) limited user

Figure 7: *Top two rows: "best proxy" IULR (right) compared to our results (left) for novel views. From top to bottom, left to right: Castle-P30, Street-10, Aquarium-20, Tree-18. Last row: Yellowhouse-12: ours, "same-size" and "best" proxy.*

interaction and silhouette-aware warp enable greatly reduced artifacts compared to previous methods, especially for challenging scenes, and (ii) our solution is very compact.

We believe that the introduction of *silhouette-aware* warping for IBR breaks new ground, and will hopefully have applications in other image-related tasks. In particular, we have demonstrated how to introduce content preserving discontinuities in a smooth variational warp. Our framework provides a trade-off between 2D image warping and 3D reconstruction. The method is robust to inaccurate geometric detail, since, other than the original (reliable) 3D reconstructed points, we only use geometry to compute blending weights. In contrast, existing techniques use the geometry to fetch texture from input images, making them sensitive to small geometric inaccuracies.

Limitations and Future Work. Our approach does not handle transparent or see-through objects such as railings, as our discontinuous warp expects somewhat contiguous objects. However, such transparent objects cannot be handled within any other IBR framework that uses geometry because depth estimation and surface reconstruction become very challenging for such cases. Our blending strategy favors using a single image than blending heavily. This causes occasional temporal popping especially on non-lambertian surfaces which appear different in different views. On the other hand, excessive blending alleviates popping but synthesized images are not crisp. Adaptive blending weights based on image content, in conjunction with 3D geometry is a promising research direction to address this limitation.

Our discontinuous warp works best for contiguous foreground objects, though we do demonstrate results for narrow objects such as tree trunks. Similar objects (e.g., lamp posts) can be handled more robustly with an adaptive-resolution warp mesh in regions with many silhouettes, as opposed to our uniform mesh.

Finally, there is scope for completely automating the silhouette extraction step, which would make our approach applicable to larger and more complex scenes. Our experiments with existing approaches (e.g., [SH09, HY10, AMFM11]) show that direct or obvious extensions of these methods do not provide adequate solutions. We consider this to be an interesting and self-contained research problem, since it would provide a new "multi-view" segmentation approach for wide-baseline image sets. Promising directions include the use of available depth and correspondence cues to eliminate false matches (see supplementary material), and automatic propagation of potentially manual silhouette information from one image to all other images in the dataset.

9. Acknowledgments

The authors acknowledge the support of the INRIA ARC NIEVE project, NVIDIA (Professor partnership program), Adobe Systems (research gift), Autodesk (Maya donation) and NSF award IIS-0905502.

References

[AMFM11] ARBELAEZ P., MAIRE M., FOWLKES C., MALIK J.: Contour detection and hierarchical image segmentation. *IEEE*

Trans. Pattern Anal. Mach. Intell. to appear (2011). 4, 9

[BBM*01] BUEHLER C., BOSSE M., MCMILLAN L., GORTLER S., COHEN M.: Unstructured lumigraph rendering. In *Proc. SIGGRAPH* (2001), pp. 425–432. 1, 2, 6, 7, 8

[BBPP10] BALLAN L., BROSTOW G. J., PUWEIN J., POLLEFEYS M.: Unstructured video-based rendering: Interactive exploration of casually captured videos. *ACM Trans. Graph* (July 2010), 1–11. 3

[CAA10] CARROLL R., AGARWALA A., AGRAWALA M.: Image warps for artistic perspective manipulation. *ACM Trans. Graph. 29* (July 2010), 127:1–127:9. 3

[CW93] CHEN S. E., WILLIAMS L.: View interpolation for image synthesis. In *Proc. SIGGRAPH* (1993), pp. 279–288. 3

[DCC*09] DURAND F., COHEN M., CHEN J., PARIS S., WANG J., MATUSIK W.: *The Video Mesh: A Data Structure for Image-based Video Editing*. Tech. Rep. MIT-CSAIL-TR-2009-062 (accepted to ICCP 2011), MIT CSAIL, 2009. 2, 3

[DP72] DOUGLAS D., PEUCKER T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer 10*, 2 (1972). 4

[DRE*10] DIDYK P., RITSCHEL T., EISEMANN E., MYSZKOWSKI K., SEIDEL H.-P.: Adaptive image-space stereo view synthesis. In *Vision, Modeling and Visualization Workshop* (Siegen, Germany, 2010), pp. 299–306. 2, 3

[DTM96] DEBEVEC P. E., TAYLOR C. J., MALIK J.: Modeling and rendering architecture from photographs: A hybrid geometry and image-based approach. *Proc. SIGGRAPH* (1996). 1, 2

[EDM*08] EISEMANN M., DECKER B. D., MAGNOR M., BEKAERT P., DE AGUIAR E., AHMED N., THEOBALT C., SELLENT A.: Floating Textures. *Comput. Graph. Forum (Proc. Eurographics) 27*, 2 (4 2008), 409–418. 2, 8

[FCSS09] FURUKAWA Y., CURLESS B., SEITZ S. M., SZELISKI R.: Manhattan-world stereo. In *Proc. CVPR* (2009), pp. 1422–1429. 2

[FP09] FURUKAWA Y., PONCE J.: Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 8 (2009), 1362–1376. 1, 2, 3, 7

[GAF*10] GOESELE M., ACKERMANN J., FUHRMANN S., HAUBOLD C., KLOWSKY R., DARMSTADT T.: Ambient point clouds for view interpolation. *ACM Trans. Graph. 29* (July 2010), 95:1–95:6. 3

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The Lumigraph. In *Proc. SIGGRAPH* (1996), pp. 43–54. 1, 2

[GSC*07] GOESELE M., SNAVELY N., CURLESS B., HOPPE H., SEITZ S. M.: Multi-view stereo for community photo collections. In *Proc. ICCV* (2007), pp. 1–8. 1, 2

[GSCO06] GAL R., SORKINE O., COHEN-OR D.: Feature-aware texturing. In *Proc. EGSR* (2006), pp. 297–303. 3

[HK09] HORNUNG A., KOBBELT L.: Interactive pixel-accurate free viewpoint rendering from images with silhouette aware sampling. *Comput. Graph. Forum 28*, 8 (2009), 2090–2103. 2

[HSEH07] HOIEM D., STEIN A. N., EFROS A. A., HEBERT M.: Recovering occlusion boundaries from a single image. *ICCV* (2007), 1–8. 4

[HY10] HE X., YUILLE A.: Occlusion boundary detection using pseudo-depth. In *Proceedings of the 11th European conference on Computer vision: Part IV* (Berlin, Heidelberg, 2010), ECCV'10, Springer-Verlag, pp. 539–552. 4, 9

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Proc. SGP* (2006), pp. 61–70. 1, 2, 7

[LGJA09] LIU F., GLEICHER M., JIN H., AGARWALA A.: Content-preserving warps for 3D video stabilization. *ACM Trans. Graph. 28* (2009), 44:1–44:9. 2, 3, 5, 6

[LH96] LEVOY M., HANRAHAN P.: Light field rendering. In *Proc. SIGGRAPH* (1996), pp. 31–42. 1, 2

[LHW*10] LANG M., HORNUNG A., WANG O., POULAKOS S., SMOLIC A., GROSS M.: Nonlinear disparity mapping for stereoscopic 3D. *ACM Trans. Graph. 29*, 3 (2010). 3

[LLB*10] LIPSKI C., LINZ C., BERGER K., SELLENT A., MAGNOR M.: Virtual video camera: Image-based viewpoint navigation through space and time. *Computer Graphics Forum 29*, 8 (2010), 2555–2568. 3

[LQ99] LHUILLIER M., QUAN L.: Image interpolation by joint view triangulation. In *Proc. CVPR* (1999), vol. 2. 3

[MB95] MCMILLAN L., BISHOP G.: Plenoptic modeling: an image-based rendering system. In *Proc. SIGGRAPH* (1995), pp. 39–46. 2

[MHM*09] MAHAJAN D., HUANG F.-C., MATUSIK W., RAMAMOORTHI R., BELHUMEUR P.: Moving gradients: A path-based method for plausible image interpolation. *ACM Trans. Graph. 28*, 3 (2009). 1, 3, 8

[Rin10] RINEAU L.: 2D conforming triangulations and meshes. In *CGAL User and Ref. Manual*, 3.7 ed. CGAL Ed. Bd., 2010. 6

[SCD*06] SEITZ S. M., CURLESS B., DIEBEL J., SCHARSTEIN D., SZELISKI R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR* (2006), vol. 1. 2

[SD96] SEITZ S. M., DYER C. R.: View morphing. In *Proc. SIGGRAPH* (1996), pp. 21–30. 3

[SH09] STEIN A. N., HEBERT M.: Occlusion boundaries from motion: Low-level detection and mid-level reasoning. *Int. J. Comput. Vision 82* (May 2009), 325–357. 4, 9

[SLW*08] STICH T., LINZ C., WALLRAVEN C., CUNNINGHAM D., MAGNOR M.: Perception-motivated interpolation of image sequences. In *Proc. APGV* (2008), pp. 97–106. 1, 3

[SS09] SHAMIR A., SORKINE O.: Visual media retargeting. In *ACM SIGGRAPH Asia Courses* (2009). 1, 3

[SSS06] SNAVELY N., SEITZ S. M., SZELISKI R.: Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph. 25*, 3 (2006), 835–846. 1, 2, 3

[SSS09] SINHA S. N., STEEDLY D., SZELISKI R.: Piecewise planar stereo for image-based rendering. In *Proc. ICCV* (2009), pp. 1881–1888. 2, 8

[SvHG*08] STRECHA C., VON HANSEN W., GOOL L. J. V., FUA P., THOENNESSEN U.: On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proc. CVPR* (2008). 8

[TC89] TEH C.-H., CHIN R.: On the detection of dominant points on digital curves. *IEEE Trans. Pattern Anal. Mach. Intell. 11*, 8 (Aug. 1989). 4

[Tol03] TOLEDO S.: *TAUCS: A Library of Sparse Linear Solvers, version 2.2*. Tel-Aviv University, Available online at http://www.tau.ac.il/~stoledo/taucs/, 2003. 6

[WLSL10] WANG Y.-S., LIN H.-C., SORKINE O., LEE T.-Y.: Motion-based video retargeting with optimized crop-and-warp. *ACM Trans. Graph. 29*, 4 (2010), article no. 90. 5

[ZCHM09] ZHANG G.-X., CHENG M.-M., HU S.-M., MARTIN R. R.: A shape-preserving approach to image resizing. *Comput. Graph. Forum 28*, 7 (2009), 1897–1906. 5