Volume 0 (1981), Number 0 pp. 1–7 COMPUTER GRAPHICS *forum*

Seamless Montage for Texturing Models

Ran Gal[1] Yonatan Wexler[2] Eyal Ofek[2] Hugues Hoppe[2] Daniel Cohen-Or[1]

[1]Tel-Aviv University, Tel-Aviv, Israel
[2]Microsoft, Redmond, USA

Abstract

We present an automatic method to recover high-resolution texture over an object by mapping detailed photographs onto its surface. Such high-resolution detail often reveals inaccuracies in geometry and registration, as well as lighting variations and surface reflections. Simple image projection results in visible seams on the surface. We minimize such seams using a global optimization that assigns compatible texture to adjacent triangles. The key idea is to search not only combinatorially over the source images, but also over a set of local image transformations that compensate for geometric misalignment. This broad search space is traversed using a discrete labeling algorithm, aided by a coarse-to-fine strategy. Our approach significantly improves resilience to acquisition errors, thereby allowing simple and easy creation of textured models for use in computer graphics.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Our goal is to generate a seamless texture over a surface model from photographs taken with an ordinary handheld camera. Given the surface shape and the set of views around it, texturing the model is a seemingly simple process: each point on the surface can be back-projected to one of the views to retrieve color at that point. In an ideal scenario (where the geometric model is accurate, the views are perfectly registered, and there are no reflections or highlights), the resulting texture is seamless. However, in practice the reconstructed geometry is only approximated, the views are rarely registered perfectly, and the model is not Lambertian.

Let us consider the simple doll model in Figure 1. The simplest approach is to map texture onto each triangle from the highest-quality photograph visible from it. However, even with a relatively detailed and accurate geometric model, imperfect registration of the views leads to noticeable seams (Figure 1(b)). One possible remedy is to locally blend together several images, but this would result in ghosting artifacts where the features do not align. In typical scenarios, the model geometry is more complex, and the creation of a seamless texture is challenging.

To overcome the artifacts introduced by per-triangle decisions, Lempitsky and Ivanov [LI07] develop a global optimization that favors smoothness in the texture assignment and penalizes the introduction of sharp seams. Figure 1(c) shows results of their approach, which indeed alleviates many artifacts. However, as shown in Figure 1(c), some seams often remain due to misregistration and imperfect geometry. The errors in Figure 1(b,c) suggest that it is unlikely that simple *image fusion* techniques alone (e.g. graph-cut, blending, etc.) can fully resolve the problem of broken image features. Closer inspection of the problem areas reveals that the errors are mostly translational. This suggests that locally shifting the textures may be the key to achieving seamless results.

We present a generalized optimization approach to overcome inaccuracies in camera registration and geometry reconstruction. Our technique introduces additional flexibility into the selection of the texture projection parameters. We search this broader space using a multi-label graph-cut optimization [Kol06], with a coarse-to-fine label refinement strategy. This augmented search space significantly improves the creation of seamless textures, as demonstrated in Figure 1(d).

Process overview. We take numerous photographs (typically a few dozen) around the subject model in free style, without restrictions or constraints. Of course, we aim to

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

(a) Input images (b) Greedy (c) [LI07] (d) Ours

Figure 1: *Overview of our contribution. (**a**) The input photographs. (**b**) Back-projecting the photographs onto the geometry of the doll by selecting the best image for each triangle in a greedy manner results in strong artifacts. (**c**) Optimizing the selection of images for texture coherence according to [LI07] alleviates the problem, but seams and broken texture are still noticeable. (**d**) Our texture montage results are nearly seamless.*

create a collection of photographs that cover the subject from all directions, or at least from directions in which the subject will be subsequently rendered. The set of photographs is then registered and calibrated using bundle adjustment [TMHF00] to provide a mapping between 3D points and pixels in the images, as illustrated in Figure 2(a).

Once the photographs are registered, the object geometry can be modeled. There are several effective techniques to reconstruct the surface geometry, as reviewed in Section 2. To emphasize the robustness of our texturing approach, we chose a relatively simple and straightforward method based on visual hull computation. We manually segment the object in a number of images, and apply silhouette carving [Lau94] to create a voxelized approximation of the model interior. A surface triangle mesh is then generated using marching cubes, and is smoothed and simplified to create an approximate model as shown in Figure 2.

Given the approximate registration and surface mesh, we seek to reconstruct the texture from the images. This texture is defined using a projective map from each mesh triangle to one of the images. Finding these mappings is the focus of this work (Section 3).

Solution approach. Accommodating the registration and modeling errors would seem to require a full perspective correction for the texture mapped on each surface triangle. Optimizing these perspective matrices would be costly, and moreover their many degrees of freedom would likely require expensive regularization. Instead, we are able to obtain excellent results using a simple screen-space translation per triangle. The intuition is as follows. Among the possible view rotation errors, pitch and yaw effectively result in image translation; it is only roll that presents a problem, and it is usually recovered accurately by bundle adjustment. Among the possible view translation erros, only translation along the depth presents a problem as it results in image scaling. Moreover, when the triangle mesh is fairly dense, second-order effects beyond local translation are effectively distributed over the many small triangles. Thus we leverage the fact that acquisition errors are well approximated locally by translation shifts within the images.

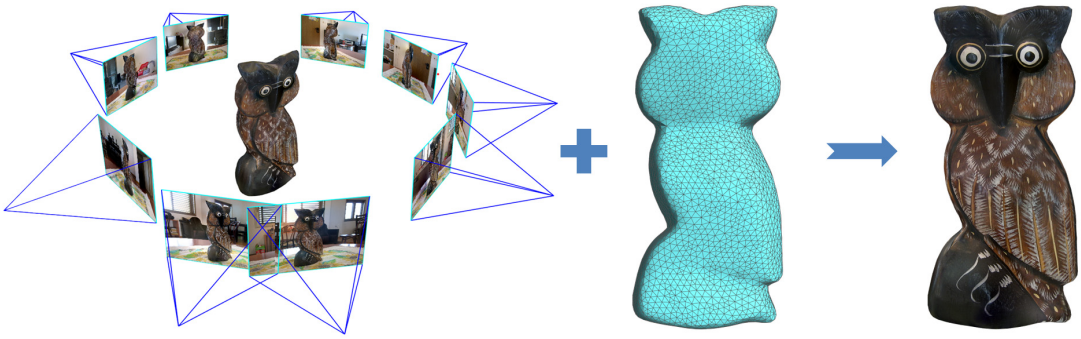
The magnitude of the needed translations can also be bounded by the reprojection errors provided by the calibration (via 3D reprojection errors) and modeling phases (via reconstruction accuracy). After globally optimizing the texture mappings, we apply a gradient-domain method on the surface mesh to attenuate color differences between adjacent projected images.

2. Previous Work

Image-based modeling has been researched extensively over the last decade [Dow02]. It involves a sequence of challenging tasks including calibration, registration, reconstruction, and texturing. Each of these steps has its own methodology and adds some degree of uncertainty and error. Our work focuses mainly on the texturing task, and accordingly also the overview in this section.

Calibration. Advanced calibration techniques are based on bundle adjustment, which simultaneously estimates the camera parameters and the positions of a sparse set of 3D points [TMHF00]. Such techniques have evolved to account for various distortions like those present in consumer cameras. Per-image calibration can be further improved using object silhouettes and image matching [LH01].

Reconstruction. From the calibrated images, surface reconstruction creates a more complete description of the



(a) (b) (c)

Figure 2: *Overview of the method. Given a set of calibrated images and a triangle mesh model, we compute a seamless texture for it. The examples shown in this paper use standard point-and-shoot camera for the source imagery. The model is either downloaded from the web or approximated via silhouette-carving. The output is a texture map for the given object.*

object shape, in the form of a dense triangle mesh. A standard approach is multi-view stereo, which infers depth by finding matching neigbhorhoods in the source images [SCDS06]. Some techniques are based on volumetric reconstruction [Dye01]. The visual hull [Lau94] is the 3D intersection of extruded image silhouettes, and can be computed quickly and efficiently. A refinement is space carving [KS98] which successively removes voxels that project to different colors in the images.

Texturing. Our focus is on texturing the reconstructed shape from the input images. As each image covers only part of the object, one must combine the sources to form a single coherent texture. Several techniques form the texture by blending the available images [WKSS01, Bau02], with the goal of finding per-texel blend weights that produce the best merged texture. One drawback of blending is that it can lead to ghosting and blurring artifacts when the textures are geometrically misaligned.

To reduce such artifacts, Lempitsky and Ivanov [LI07] pose texturing as an image stitching problem. Each surface triangle is projected onto the images from which it is visible, and its final texture will be assigned entirely from one image in this set. The goals are to select the best texture source and to penalize mismatches across triangle boundaries. This results in a Markov Random Field problem. Lighting variations are corrected as a postprocess using a piecewise continuous function over the triangles. Our approach builds on this work, with two major differences. We expand the combinatorial search to consider local image translations; this change lets the technique compensate for calibration and reconstruction errors, and proves remarkably effective. Second, to eliminate the residual lighting variations, we rely on Poisson blending [PGB03] in the texture image domain.

Interestingly, the approach of locally shifting texture content per-triangle has been explored previously in the context of texture synthesis, by Soler et al. [SCA02]. Given a texture exemplar, a seamless texture is formed over a surface mesh by optimizing a set of per-face imaging transformations. Their approach works in a greedy manner rather than with a global MRF. Eisemann et al. [EDM*08] deal with the misalignment by computing pairwise warping between the images using optical flow. These are then fused at rendering time. Both the computation and the rendering rely on the GPU. When one coherent texture can be prepared ahead of time, the rendering effort is greatly simplified both in term of storage and computation as only one texture is needed. When the object and camera calibration are indeed accurate, Super-Resolution methods may be applied on the textures as well as is suggested by [GC09].

Several recent techniques provide semi-automated tools to allow user placement of texture content onto shapes, e.g. [KSG03, ZWT*05, SGW06, TT09]. These techniques let the user specify a number of matching points between the triangle mesh and an image region, and solve a relaxation problem to find a low-distortion map onto the surface. Our work differs in that it automatically selects content from among multiple photographs, and most significantly, it addresses the problem of minimizing seams between the mapped regions.

Thormählen et al. [TS08] address the problem of creating "ortho-images" (orthographic views from orthogonal directions) given a set of calibrated images of an object. This process can be viewed as that of texturing an extremely simplified object, namely an axis-aligned box.

3. Image-based texturing

Overview. Our core strategy for texturing builds on the approach of Lempitsky and Ivanov [LI07]. The texture on each mesh triangle is defined by projecting the triangle face onto

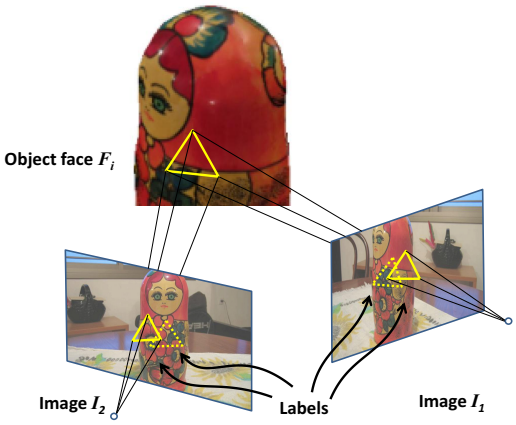


Figure 3: *Mapping from images to a surface triangle. Given errors in the surface location or the camera parameters, the default recovered projection is inaccurate. Our solution is to allow translation shifts within the images.*

one of the input images $\{I_1, \ldots I_N\}$. This projection between faces and images uses the camera parameters and the mesh geometry.

We seek to assign a label to each face that identifies its associated input image. This label assignment has two goals: (1) selecting texture content with high resolution, low anisotropy, and high contrast, and (2) forming seamless texture joins between adjacent triangles.

Our contribution is to allow additional degrees of freedom into the texture projections to compensate for inaccuracies in the image acquisition, bundle adjustment, and particularly the geometric model. This additional flexibility results in significantly fewer seam artifacts in the final surface texture montage.

Specifically, we expand the space of face labels to include an image-space transformation, which is applied after the projection in image space. As discussed in Section 1, we let this transformation be a simple translation. The label associated to each triangle face F_i thus consists of a tuple $l_i = (s_i, t_i)$ where $s_i \in \{1 \ldots N\}$ identifies the source image, and $t_i \in \mathbb{R}^2$ is a 2D translation vector. This tuple defines a perspective map ϕ_{l_i} from mesh face F_i into image I_{s_i}.

Finding a good label assignment is therefore a hybrid combinatorial continuous optimization. Our approach is to make the optimization purely combinatorial by discretizing the translation vectors. We next present our scheme in further detail.

Objective function. Our optimization has the form

$$\min_{l_1 \ldots l_N} \sum_{i=1}^{N} E_{\text{data}}(l_i) + \lambda \sum_{\{i,j\} \in M} E_{\text{smooth}}(l_i, l_j). \quad (1)$$

In the first term,

$$E_{\text{data}}(l_i) = -\delta_i \int_{\phi_{l_i}(F_i)} \|\nabla(I_{s_i}(p))\|^2 \, dp \quad (2)$$

seeks to texture each face from the "best" image that sees it. It does so by integrating the total image variation mapped onto the triangle, in image space. Note that this definition incorporates the effect of foreshortening (as a smaller region would be integrated), image resolution, and blur. The normalization constants δ_s are the ratios between triangle perimeter and area.

In the second term,

$$E_{\text{smooth}}(l_i, l_j) = \int_{\mathcal{N}(F_i \cap F_j)} \left\| I_{s_i}(\phi_{l_i}(p)) - I_{s_j}(\phi_{l_j}(p)) \right\|^2 dp \quad (3)$$

measures the texture discrepency between adjacent triangles. It seeks textures on adjacent faces whose colors agree in a neighborhood of their shared edge. Note that this term is zero for pairs of faces with the same label. Typically, the natural result of the optimization will be a set of mesh charts, each containing faces with similar or identical labels, so that most of the error is concentrated on the optimized boundaries between the charts.

Both energy term have the same unit of measure, as they both integrate the squared difference of pixel values over the perimeter of the triangles.

Optimization.

The objective function (1) forms a Markov Random Field (MRF) problem, in which the graph nodes correspond to the mesh faces and each node has a discrete set of possible labels. We solve this MRF problem using the method of [Kol06].

The number of labels that result from our formulation is often beyond the capabilities of current implementations. We therefore perform the optimization in an iterative coarse-to-fine scheme to allow the recovery of precise translations without having to consider an excessively large combinatorial space. The images are coarsened in a Gaussian pyramid to half the resolution in each dimension. In the coarsest iteration, the set of candidate labels for each face consists of all N possible image assignments times 9 possible translation vectors $\{-1, 0, +1\}^2$ in pixel units at that pyramid resolution. The result of the optimization provides the best label assignment l_i^* for each node. Rather than limiting ourselves to this single choice, we rank all other labels for l_i based on (3) as the sum $\sum E_{\text{smooth}}(l_i, l_j^*)$ where the graph neighbors of node i are assigned their optimal labels l_j^*. The top entries in this ranking are used to seed the candidate labels at the next finer

Figure 4: *The results before and after applying color correction using Poisson blending.*

Dataset name	Input photographs	Mesh triangles	Processing time (minutes)
Doll	9	12,000	15
Striped cat	8	13,500	15
Monkey	12	15,000	15
Brown cat	15	18,500	30
Yellow car	16	17,000	25
Tall cat	10	11,000	7

Table 1: *Quantitative results for the various examples.*

level. Specifically, each label entry is considered in conjunction with 9 new shift vectors. This allows the hierarchical optimization to maintain several possible candidates.

We perform three MRF iterations at each shift size, to allow larger total translations. The coarsest pyramid level is chosen to accommodate the magnitude of the registration and reconstruction errors (e.g., six pyramid levels to allow an error of ± 64 pixels). For greater quality, the finest pyramid level is solved again with shifts of $1/2$ pixels. While the graph-cut-based MRF optimization method is guaranteed to converge, it may not converge to the global optimum.

Poisson blending.

The discrete optimization can deal with large errors. One effect that remains after textural alignment is local variations in luminosity. These variations are due to the automatic gain of the camera as well as surface reflections. These create small but noticeable artifacts as can be seen in Figure 4(left).

We resolve these using the state-of-the-art method of [PGB03], applied to the input textures. In each image, contiguous texture patches are identified. A 1-pixel wide boundary around each patch is identified and adjusted halfway towards the pixel values on the other side of the boundary as induced by the solution. This ensures a good fit and provides the necessary boundary conditions for the interior of the patch. Solving this step for pixels in image space (rather than for mesh vertices) lets us leverage algorithms optimized for solving Poisson equations on regular grids. We found that each patch can be solved independently without incurring artifacts.

4. Implementation and results

We implement our method on commodity hardware (3 GHz processor with 3 GB RAM). In a typical use, we take 20–30 photos around the object using a consumer point-and-shoot camera (6-megapixel resolution). These photos are then calibrated via bundle adjustment and every other photo is taken, resulting in 10–15 photos. The visual hull is computed from the segmentation of the object in 5–6 of these. The hull mesh is simplified to 10–15 K triangles (Table 1). Our coarse-to-fine MRF optimization is performed over 6 pyramid levels. Altogether the process takes under an hour from start to finish, with 5 minutes for capture, 2 for copying the images, 1 for calibration, 10 for segmentation, and 5–30 for the optimization. The output is a set of projective textures for the object. Figures 5–11 show a varitey of example results, and highlight benefits and limitations.

Figure 9 shows a textured car model, where the visual hull mesh is a rather crude approximation of the car shape. Nonetheless, our method is able to generate a plausible smooth texture (Figure 9(b)). Figures 5, 6, 8, 10, and 11 show the results of our method using images of different objects. Note the sparsity of texture seams even in the presence of inaccurate geometry. Figure 7 shows the reconstruction of texture for the same model with different levels of geometric approximation. Note the nice seamless texture despite the geometric simplification.

The example in Figure 11 highlights some limitations of our scheme. In the presence of high-frequency texture content and significant geometric errors, it is unable to form a seamless montage. In some other cases, a seamless result is obtained, but at the expense of severe warping distortion. For instance, the features in the monkey face of Figure 6(d) are distorted compared to the input image in Figure 6(a).

5. Conclusions

Texturing is one of the more time-consuming stages in a typical modeling pipeline. We present a practical, fully automated technique for seamlessly texturing a 3D model from a set of casual photographs. In particular, our method robustly handles inaccuracies in input including shape and alignment errors. The key insight is the realization that most texture artifacts can be eliminated through local image-space translations. The result is one texture for the whole object which minimizes visual artifacts. This simple strategy proves remarkably effective.

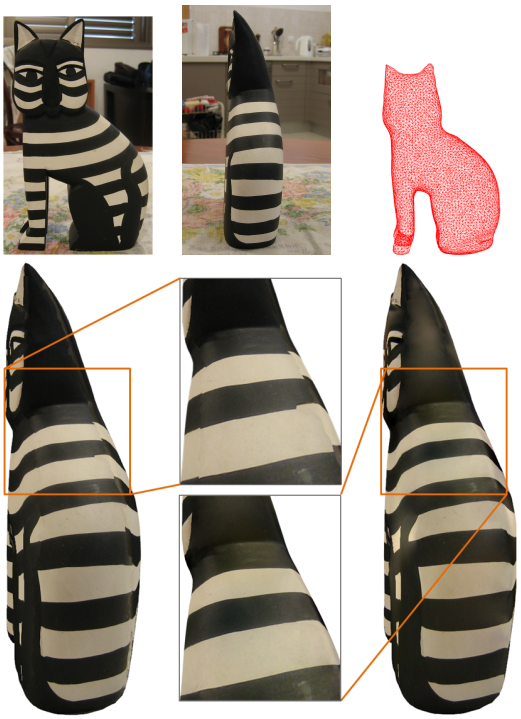


Figure 5: *Example demonstrating the benefit of optimizing with translation shift vectors. Top row: two input images and reconstructed mesh. Bottom row: resulting texture montage without and with shift optimization.*

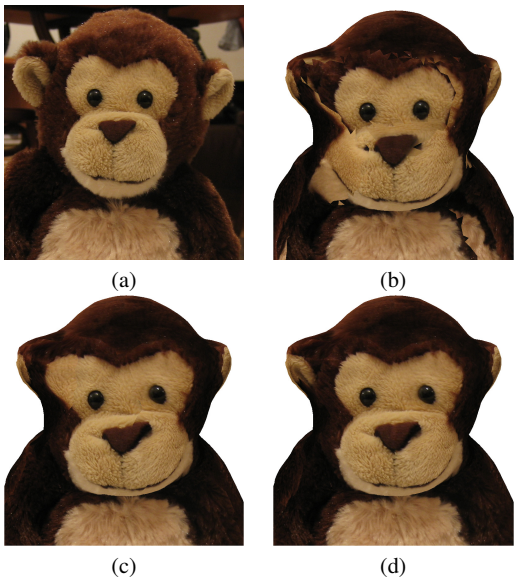


Figure 6: *Monkey. (a) One of the input images. (b) Simple back-projection. (c) The optimization without shifts is able to reduce seams but at the cost of severe distortions. (d) Allowing the mapping translations eliminates the distortions and produces a seamless texture.*

References

[Bau02] BAUMBERG A.: Blending images for texturing 3D models. In *Proc. Conf. on British Machine Vision Association* (2002), pp. 404–413. 3

[Dow02] DOWNING G.: Image based modeling: Essentials. 3D modeling from photographs with Greg Downing. Gnomon Workshop, 2002. 2

[Dye01] DYER C. R.: Volumetric scene reconstruction from multiple views. In *Foundations of Image Understanding* (2001), Kluwer, pp. 469–489. 3

[EDM*08] EISEMANN M., DECKER B. D., MAGNOR M., BEKAERT P., DE AGUIAR E., AHMED N., THEOBALT C., SELLENT A.: Floating Textures. *Computer Graphics Forum (Proc. Eurographics EG'08) 27*, 2 (4 2008), 409–418. 3

[GC09] GOLDLUECKE B., CREMERS D.: Superresolution texture maps for multiview reconstruction. In *IEEE International Conference on Computer Vision (ICCV)* (Kyoto, Japan, 2009). 3

[Kol06] KOLMOGOROV V.: Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell. 28*, 10 (2006), 1568–1583. 1, 4

[KS98] KUTULAKOS K. N., SEITZ S. M.: A theory of shape by space carving. *International Journal of Computer Vision 38* (1998), 307–314. 3

[KSG03] KRAEVOY V., SHEFFER A., GOTSMAN C.: Matchmaker: constructing constrained texture maps. *ACM Trans. Graph. 22*, 3 (2003), 326–333. 3

[Lau94] LAURENTINI A.: The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell. 16*, 2 (1994), 150–162. 2, 3

[LH01] LENSCH H. P. A., HEIDRICH W.: A silhouette-based algorithm for texture registration and stitching. *Graphical Models 63* (2001), 245–262. 2

[LI07] LEMPITSKY V., IVANOV D.: Seamless mosaicing of image-based texture maps. *Computer Vision and Pattern Recognition 0* (2007), 1–6. 1, 2, 3

[PGB03] PÉREZ P., GANGNET M., BLAKE A.: Poisson image editing. *ACM Trans. Graph. 22*, 3 (2003), 313–318. 3, 5

[SCA02] SOLER C., CANI M.-P., ANGELIDIS A.: Hierarchical pattern mapping. *ACM Trans. Graph. 21*, 3 (2002). 3

[SCDS06] SEITZ S. M., CURLESS B., DIEBEL J., SZELISKI D. S. R.: A comparison and evaluation of multi-view stereo reconstruction algorithms, 2006. 3

[SGW06] SCHMIDT R., GRIMM C., WYVILL B.: Interactive decal compositing with discrete exponential maps.

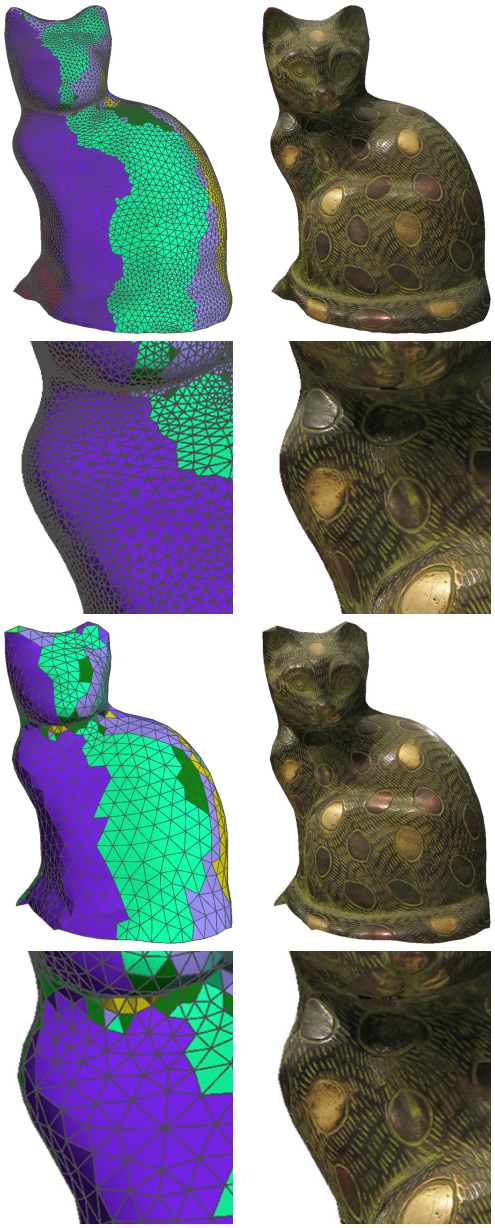


Figure 7: *Comparison of texture montages on different geometries. The top result uses a fine mesh with 8,000 triangles, whereas the bottom result uses a coarse mesh with 1,200 triangles. The reconstructed texture remains seamless, despite the significant geometric errors in the surface such as those evident near the ears.*

Figure 8: *Additional examples, showing some of the input images, the reconstructed mesh colored by labels, and some resulting textured views.*

ACM Trans. Graph. 25, 3 (2006). 3

[TMHF00] TRIGGS B., MCLAUCHLAN P., HARTLEY R., FITZGIBBON A.: Bundle adjustment – a modern synthesis. *Lecture Notes in Computer Science 1883* (2000), 298–372. 2

[TS08] THORMÄHLEN T., SEIDEL H.-P.: 3D-modeling by ortho-image generation from image sequences. *ACM Trans. Graph. 27*, 3 (2008), 1–5. 3

[TT09] TZUR Y., TAL A.: Flexistickers : Photogrammetric texture mapping using casual images. *ACM Trans. Graph. 28*, 3 (2009). 3

[WKSS01] WANG L., KANG S. B., SZELISKI R., SHUM H.-Y.: Optimal texture map reconstruction from multiple views. In *Computer Vision and Pattern Recognition* (2001). 3

[ZWT*05] ZHOU K., WANG X., TONG Y., DESBRUN M., GUO B., SHUM H.-Y.: TextureMontage. *ACM Trans. Graph. 24*, 3 (2005), 1148–1155. 3

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

Figure 9: *Texturing a yellow car. (a) One of the casual photographs, and the labeled mesh generated by the optimization. (b) Two views of the textured model. (c) Close-up view showing the benefit of adding shifts to the optimization; note the broken lines on the hood.*

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

Figure 10: *Tall cat. (a) One photograph of the cat model. (b) Optimization without shifts. (c) Optimization with shifts. (d) Close-up views.*

(a) (b) (c)

Figure 11: *Brown Cat. (a) A few of the input photographs and the generated labeled mesh. (b) Three renderings of the textured model. There is one small misalignment above the cat's leg in the light blue square. (c) Close-up views, top and bottom, without shift and with shifts. Note that in the rightmost example, the shifts do not successfully recover a seamless texture, due to large errors in the geometry.*