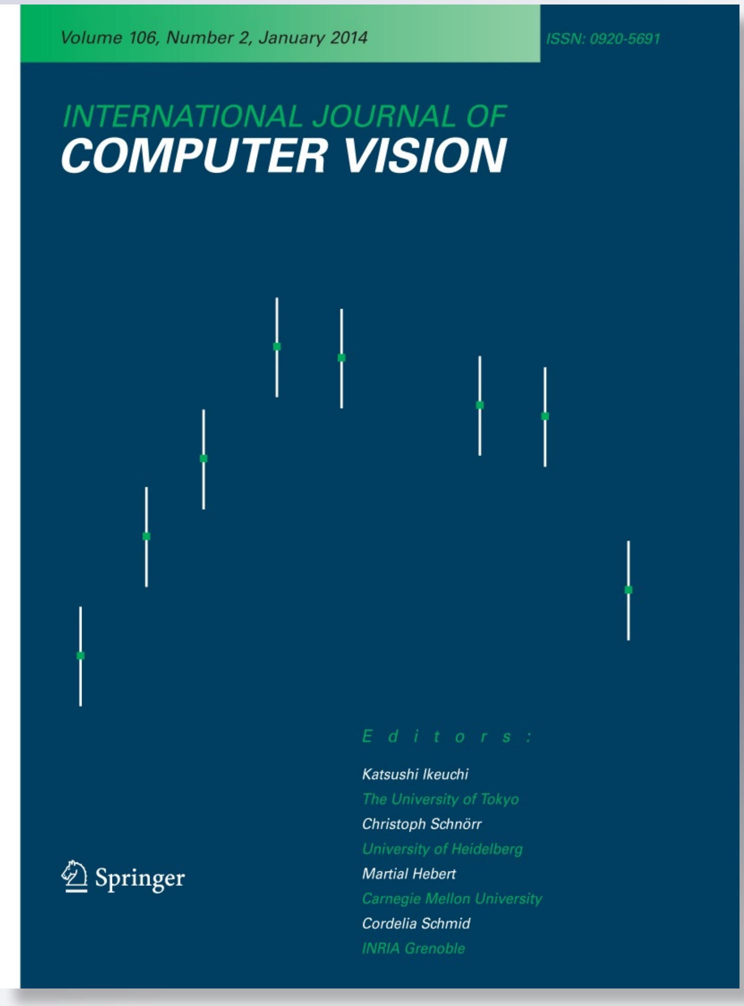
A Super-Resolution Framework for High-Accuracy Multiview Reconstruction

Bastian Goldlücke, Mathieu Aubry, Kalin Kolev & Daniel Cremers

International Journal of Computer Vision

ISSN 0920-5691
Volume 106
Number 2

Int J Comput Vis (2014) 106:172-191
DOI 10.1007/s11263-013-0654-8



Springer

Int J Comput Vis (2014) 106:172–191
DOI 10.1007/s11263-013-0654-8

A Super-Resolution Framework for High-Accuracy Multiview Reconstruction

Bastian Goldlücke · Mathieu Aubry · Kalin Kolev · Daniel Cremers

Received: 23 May 2012 / Accepted: 12 August 2013 / Published online: 25 August 2013

Abstract We present a variational framework to estimate super-resolved texture maps on a 3D geometry model of a surface from multiple images. Given the calibrated images and the reconstructed geometry, the proposed functional is convex in the super-resolution texture. Using a conformal atlas of the surface, we transform the model from the curved geometry to the flat charts and solve it using state-of-the-art and provably convergent primal–dual algorithms. In order to improve image alignment and quality of the texture, we extend the functional to also optimize for a normal displacement map on the surface as well as the camera calibration parameters. Since the sub-problems for displacement and camera parameters are non-convex, we revert to relaxation schemes in order to robustly estimate a minimizer via sequential convex programming. Experimental results confirm that the proposed super-resolution framework allows to recover textured models with significantly higher level-of-detail than the individual input images.

B. Goldlücke (✉)
Heidelberg Collaboratory for Image Processing,
University of Heidelberg, Heidelberg, Germany
e-mail: bastian.goldluecke@iwr.uni-heidelberg.de

M. Aubry · D. Cremers
Computer Science Department, Technical University of Munich,
Munich, Germany
e-mail: mathieu.aubry@in.tum.de

D. Cremers
e-mail: cremers@tum.de

K. Kolev
Department of Computer Science, ETH Zurich, Zurich, Switzerland
e-mail: kalin.kolev@inf.ethz.ch

Keywords Multi-view 3D reconstruction · Texture reconstruction · Super-resolution · Camera calibration · Variational methods

1 Introduction

Modern image-based 3D reconstruction algorithms achieve high levels of geometric accuracy. However, due to intrinsic limitations like voxel volume resolution and mesh size, the geometric resolution of the model is usually well below the pixel resolution in a rendering. This leads to a number of problems if one wants to estimate a high-resolution texture for the model from the camera images. Most importantly, since geometry is never perfectly accurate on the level of individual texels, the image registration cannot be exactly correct, which leads to a blurry estimated texture, see Fig. 1. Consequently, previous methods on texture generation usually employ some form of additional registration before estimating texel color (Bernardini et al. 2001; Lensch et al. 2001; Theobalt et al. 2007).

In methods fitting a local appearance model on a per-texel basis, it is generally true that the fewer source cameras influence the result for a single texel, the sharper the resulting texture will be. However, if only the contributions of few cameras are blended for a given texture patch, it is likely that seams and discontinuities arise at visibility boundaries, so some form of stitching has to take place to smoothen the result (Allne et al. 2008; Lempitsky and Ivanov 2007). Furthermore, not using all available source images implies discarding a lot of potentially useful information.

In particular, in multi-view settings, usually every patch of the surface is captured from several cameras. Therefore, using a suitable super-resolution model, one should be able to recover the texture map in higher resolution than provided by

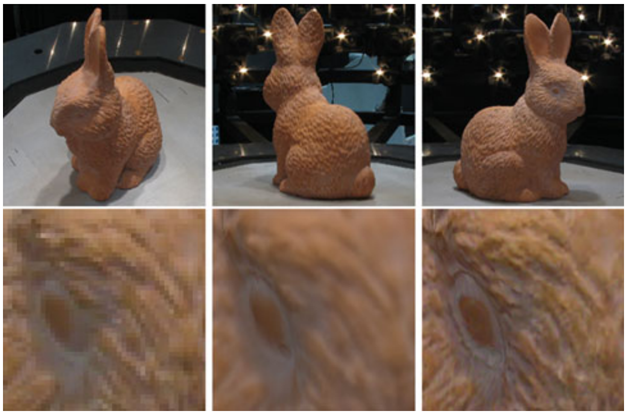


Fig. 1 *Top* multi-view input images for the reconstruction. *Bottom left* close-up of one of the low-resolution input images. *Center* rendered model with blurry texture initialized by weighted averaging of input images. *Right* high-quality texture optimized with the proposed super-resolution approach

the input images. However, this possibility has not yet been explored for the curved 3D models obtained by 3D reconstruction methods, with most existing methods fitting a local lighting model per-vertex or per-texel only, disregarding texel interdependencies. Thus, our paper aims at opening up the highly interesting area of super-resolution models on surfaces.

The super-resolution framework presented in this paper is designed to address all of the shortcomings mentioned above. We account for the interdependency of geometry and photometry by minimizing a single functional with respect to all relevant unknowns: a super-resolved texture map for the surface, a displacement field optimizing local surface geometry, as well as camera calibration parameters. The image formation model is based on the current state-of-the-art (Fransens et al. 2007; Schoenemann and Cremers 2008; Sroubek et al. 2007), for which there is a well-developed theory (Baker and Kanade 2002). We are able to recover the texture in much higher resolution and level-of-detail than provided by the input images. By design, the method scales very well with the number of input cameras: more cameras will always lead to a more accurate solution. The resulting models are of excellent quality and can be rendered from arbitrary viewpoints.

1.1 Contributions

We introduce the first framework for texture super-resolution on curved surfaces, which recovers the texture as the solution to a variational inverse problem. It unifies the problems of photometric and geometric reconstruction, since the very same model can be optimized with respect to different variables. The quality of texture reconstruction is mainly dependent on a good alignment of the source images on the model. In our framework, this is achieved by optimizing for a displacement map on the surface to improve the local geometry estimates, as well as by optimizing for the camera calibration parameters.

Providing a working framework for the different minimization problems is in each case highly non-trivial. We show how to handle variational models on curved surfaces in practice, including automatic texture atlas generation and handling of chart boundaries in order to transform the problem to 2D planar texture space. We then show how to solve the different subproblems in practice using modern convex optimization algorithms.

The proposed texture reconstruction algorithm has a number of specific advantages compared to previous approaches such as Bernardini et al. (2001), Lensch et al. (2001), Theobalt et al. (2007). In particular,

- An arbitrary number of source images can be integrated in a natural way, globally reconstructing a spatially coherent texture map. No inherent scaling problems arise, in fact, we embrace having more cameras as this leads to a more accurate solution.
- Suitable weighting factors for input image contributions arise naturally, and do not have to be imposed by heuristic assumptions.
- Neither visibility boundaries nor seams have to be treated explicitly, since the regularizer in the energy already minimizes discontinuities.

We also revisit variational camera calibration in a spatially dense setting. Compared to the work of Unal et al. (2007), we introduce a novel optimization algorithm which decouples the estimation of point correspondences and camera parameters. In this manner, we show that the original minimization problem can be solved by alternating dense correspondence estimation using essentially optical flow with camera parameter estimation given by a spatially continuous version of bundle adjustment. In experiments, we demonstrate that the optimized camera calibration can provide substantial improvements in the estimated texture.

The present work extends and consolidates our previous conference publications (Aubry et al. 2011; Goldluecke and Cremers 2009a,b). We have unified all notation and reworked the exposition of the theory into an extended, easily accessible form. In addition, we have modernized the algorithms from the two earlier papers (Goldluecke and Cremers 2009a,b) by switching from PDE-based gradient descent schemes to recent convex optimization methods (Chambolle and Pock 2011; Goldluecke et al. 2012), which we also have extensively tested in a recent related work on light field super-resolution Wanner and Goldluecke (2012). In effect, the proposed framework for texture super-resolution becomes both easier to implement as well as computationally more efficient. We have made a generic implementation for solving the super-resolution inverse problem available online.[1]

2 Related Work

The problem of multi-view 3D reconstruction is one of the most fundamental and extensively studied problems in computer vision. Following recent improvements in digital photography, it has undergone a revolution in recent years and is approaching the accuracy of the most reliable techniques for 3D modeling (Seitz et al. 2006; Strecha et al. 2008). The vast branch of works on image-based modeling can roughly be classified into three groups with respect to the utilized surface representation: implicit (volumetric), mesh-based and sparse (or quasi-dense) by means of a point cloud (possibly oriented). As the proposed framework is independent of the particular choice of initial geometry estimate, we will not consider these methods in detail here, but refer to the detailed taxonomy in Seitz et al. (2006), Strecha et al. (2008).

At the core of each multi-view reconstruction pipeline is the calibration of the cameras, i.e. the estimation of position, orientation and intrinsic parameters for each camera. In the last two decades, great efforts have been focused on automatic camera calibration based on image information alone. As a result, we now have a number of publicly available software packages by Klein and Murray (2007) and Snavely et al. (2006), which allow to automatically determine the camera parameters from a collection of images. Yet, in the context of image-based modeling, the question of where each camera was located is obviously tightly intertwined with the estimation of geometry and texture. A highly accurate estimate of the object's geometry and texture should help to further improve the estimation of camera parameters. Analogously, geometry and texture modeling could benefit from a more precise camera calibration.

Up to date, it could be observed that while geometry and color are reconstructed in a *dense* manner, camera calibration methods typically rely on *sparse* feature correspondences. As the calibration problem is highly overdetermined (only 11 parameters are to be estimated for each camera in our setting), a straightforward way to address it is to robustly pick a small subset of the provided information (*feature* points) for the estimation process.

This naturally leads to the investigation of sparse feature-point based methods (Hartley and Zisserman 2004) which have become an established tool. Yet, the accuracy of the obtained parameters strongly depends on the precision of the underlying feature-point detector as well as the reliability of the matching procedure. Even though there exist multiple ways to tackle these tasks, like considering epipolar constraints and robust model fitting, the exploration of dense formulations to better understand the nature of the registration process deserves more attention.

2.1 Camera Calibration

A key ingredient of a multi-camera calibration system is a global optimization step involving all camera parameters and estimated sparse 3D geometry, called *bundle adjustment* (Triggs et al. 2000). Usually, the calibration pipeline is split into multiple sequential stages: first, feature points and corresponding descriptors are estimated, then, an initial matching and 3D structure are established, and finally, a global refinement step is performed while filtering out mismatches and outliers. In practice, the precision of the underlying feature points and respective descriptors is limited, since the computations are performed on a pixel basis without any knowledge of the observed geometry. In this work, we show how estimated dense 3D structure can be exploited to further improve a given calibration.

Recently, Furukawa and Ponce (2009) suggested a stereo-based approach for calibration refinement. The method starts with initial calibration parameters and a sparse 3D point cloud representing the observed geometry and assigns an oriented patch to each point. The optimal calibration parameters and the precise localization of each 3D point in space are obtained by finding the local orientation giving rise to the most consistent patch distortion. Although this approach significantly improves upon classical sparse methods, it is still limited by the underlying local planarity assumption.

The refinement of camera parameters in spatially dense reconstruction methods has been further generalized in the variational approach of Unal et al. (2007). There, the authors suggest a generative model of image formation and subsequently estimate both intrinsic parameters (focal length and skew) and extrinsic parameters (translation, rotation) by a gradient descent approach. However, the proposed formulation is based on particular assumptions regarding the radiance of the 3D scene—piecewise smoothness for the object and constancy for the background.

A closer analysis of this latter approach reveals that the camera parameters are estimated based on minimizing a *photometric* error (color difference between modeled and observed scene). Interestingly, this is fundamentally different from the *geometric* error (reprojection error) that most sparse state-of-the-art algorithms like bundle adjustment minimize. In this paper, the problem of refining camera calibration parameters is embedded in a spatially dense multi-view reconstruction framework. The decoupling strategy we propose for optimization gives rise to a spatially dense version of bundle adjustment.

[1] http://cocolib.net.

2.2 Image-Based Texture Reconstruction

In image-based texture generation, one faces the problem of integrating several views of the same surface region into a single texture map. In most approaches to date, information from the cameras is blended per texel, using a suitable heuristic to weight contributions from different source images (Bernardini et al. 2001; Lensch et al. 2001; Theobalt et al. 2007). It is common to perform additional image registration before estimating surface properties, in order to correct for small-scale geometry errors (Bernardini et al. 2001; Eisemann et al. 2008) or object motion (Theobalt et al. 2007). Other algorithms circumvent texel-wise blending by selecting large regions which are assigned texture data from a single camera. However, those methods then face the problem of stitching together the different regions. Optimal seam locations can be estimated beforehand (Lempitsky and Ivanov 2007), in combination with multi-band blending at seam locations (Allne et al. 2008).

Although existing methods produce visually pleasing results, there is one problem intrinsic to the approach of selecting optimal cameras and blending the contributions. Namely, this technique does not scale favourably with the number of input cameras. Indeed, the result from blending becomes more blurred the more images are blended together, in particular if geometry estimates are not perfectly accurate. Thus, many algorithms end up throwing away most information in order to increase sharpness of the results. However, the fewer cameras contribute to a single patch, the more the visible seams reappear at patch boundaries. These must subsequently be dealt with by postprocessing.

The proposed approach of texture super-resolution delivers a solution to this dilemma. It is based on previous models employed in 2D super-resolution (Capel and Zisserman 2001; Fransens et al. 2007; Schoenemann and Cremers 2008; Sroubek et al. 2007) for which there is a well-developed theory (Baker and Kanade 2002). Of course, some other approaches to super-resolution also go beyond simply matching 2D images. Related is the work by Bhat et al. (2007), who employ multi-view stereo to obtain a 3D model for a video sequence, which is then used for example to transfer information from photographs to frames in the video. By providing additional high-resolution images as input, they can effectively create a super-resolved video, although they neither employ an image formation model for super-resolution nor solve an inverse problem for it. A recent work on super-resolution and novel view synthesis in light fields (Wanner and Goldluecke 2012) is inspired by the framework proposed in this paper. The difference is that the transformation between views is given by a disparity map, and there is no intermediate surface texture reconstructed to generate new views. Instead, an inverse problem for the super-resolved novel view is solved directly. A similar image formation model is used by Rav-Acha et al. (2008). They compute a texture map for a deforming surface in a video sequence, which they call an *unwrap mosaic*. The unknown mapping from texture space to image space of each frame is recovered in an energy minimization framework. Notably, they also account for visibility and foreshortening effects using the Jacobian determinant. While they do not recover a textured 3D model such as we do, by changing the texture they can make efficient editing operations in the full video.

Our model generalizes previous approaches to super-resolution to curved surfaces in 3D space. As usual, we describe an image formation process which takes into account the behavior of the sensor elements of the cameras. A texture on the surface is then estimated to optimally fit all input images simultaneously. All images are treated symmetrically, and there is no inherent disadvantage in using more input images. In fact, more images give rise to even sharper texture estimates. Seams are taken care of implicitly by weighting factors which follow naturally from the model. Due to these advantages, the estimated texture maps are of excellent quality and contain more details than each single input image taken by itself, see Fig. 1.

3 Variational Texture Super-Resolution

In this section, we introduce a super-resolution image formation model, which depends on the surface texture map, geometry and camera calibration parameters. Subsequently, we will show how it can be efficiently optimized with respect to all these unknowns. However, in order to bootstrap the process, we require that an initial estimate for the surface geometry and camera calibration is available. In the paper, we use a combination of bundler (Snavely et al. 2006) and the variational 3D reconstruction methods proposed in Kolev et al. (2009, 2010). The initial texture map will be computed by the algorithm presented here.

In the following, let $\mathcal{I}_1, \ldots, \mathcal{I}_n : \Omega_i \to \mathbb{R}$ be the input images captured by cameras with projections $\pi_1, \ldots, \pi_n : \mathbb{R}^3 \to \mathbb{R}^2$, see Fig. 2. The cameras observe a known Lambertian surface $\Sigma \subset \mathbb{R}^3$, which is textured with the unknown texture map $T : \Sigma \to \mathbb{R}$. For the sake of simplicity of notation, we will for now assume grayscale images. The extension to color textures is straightforward and requires just an adaption of the regularizer, see Sect. 4.4.

In the remainder of the section, we will first show how to recover the unknown texture. Later, we will extend the model to also locally optimize the geometry Σ and projection parameters $\pi_1, \ldots, \pi_n$ by considering them as additional variables in the energy.

3.1 Image Formation and Resulting Inverse Problem

The basic idea is to recover the unknown texture map as the solution to an inverse problem, which is formulated as the minimization of an energy functional consisting of a data term and a regularization term,

$$E(T) := E_{\text{data}}(T) + \lambda E_{\text{tv}}(T), \quad \text{with}$$
$$E_{\text{tv}}(T) := \int_{\Sigma} \|\nabla_{\Sigma} T\|_{\Sigma} \, \mathrm{d}s. \tag{1}$$

For the regularizer, we choose the total variation semi-norm of the texture map, since it is well-suited to preserve a crisp texture with sharp high-resolution features. In the above equation, $\lambda > 0$ is a parameter controlling the desired smoothness of the result. It is usually set to a small value, since its main purpose is to improve convergence speed and stability of the optimization scheme, while an actual smoothing contribution is undesireable for accurate textures.

Note that the total variation is defined by an integration *over the surface* Σ, so technically we require the surface gradient operator ∇_{Σ} (measuring variation of T along the surface) and tangent space norm. These are explained in detail in Lui et al. (2005). For implementation of the minimization process, however, it is completely sufficient to understand the operators after transformation to the texture space. This is given later in Eq. (9).

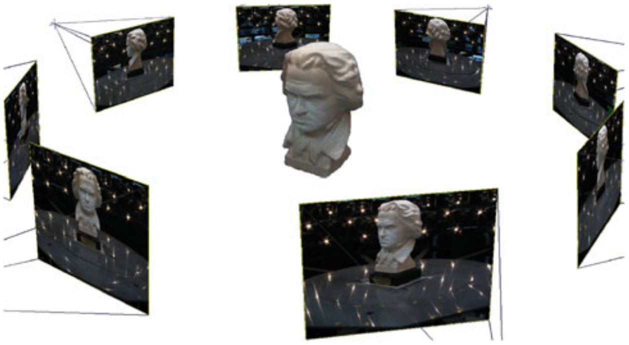


Fig. 2 Input images and geometry. Multiple cameras capture a static object. Each surface patch is visible in several input images, enabling texture super-resolution

The core of the model is the data term, which is based on a current state-of-the-art super-resolution formulation (Sroubek et al. 2007). The idea is that a real-world camera downsamples the input by integrating over the visible texels inside each sensor element, see Fig. 3. This integration process is modeled with a convolution kernel b, which can be derived from the specifications of the camera (Baker and Kanade 2002). Thus, in an ideal setting, if we take the texture, project it down into the image plane and compute the convolution, we should exactly obtain the input image. However, in a real-world setting, this will not be completely satisfied due to errors in projections and geometry, or image noise. The resulting data term for a MAP model under the assumption of impulse noise on the input images is

$$E_{\text{data}}(T) := \sum_{i=1}^{n} \int_{S_i} |b * (T \circ \beta_i) - \mathcal{I}_i| \, \mathrm{d}x. \tag{2}$$

We choose the L^1 norm of the difference, since we found experimentally that it is more robust with respect to outliers than the more simple to minimize L^2 model—see also Mitzel et al. (2009). The backprojection mappings $\beta_i : S_i \to \Sigma$ assign to each point inside the silhouettes $S_i := \pi_i(\Sigma) \subset \mathbb{R}^2$ the nearest point on the surface Σ, see Fig. 4. Note that they are inverse to the projections, which are one-to-one when restricted to the visible parts of Σ. The concatenation $T \circ \beta_i$ of the texture map with the backprojection thus denotes the image of the visible texture area on the image plane. It is best interpreted as a high-resolution rendering of the current model, given the current estimate for the texture. As usual, the operator '$*$' denotes convolution. Note that technically, $T \circ \beta_i$ is only defined inside the silhouette. However, to not clutter notation, we implicitly assume that it is extended with zero to the whole image domain for the purposes of computing the convolution and subtracting the input image. Furthermore, after discretization, the convolution operation implies a subsampling to the lower resolution of the input images.

3.2 Transforming The Data Term to the Surface

In order to consistently integrate data term and regularizer over the same domain, we first express the data term integrals

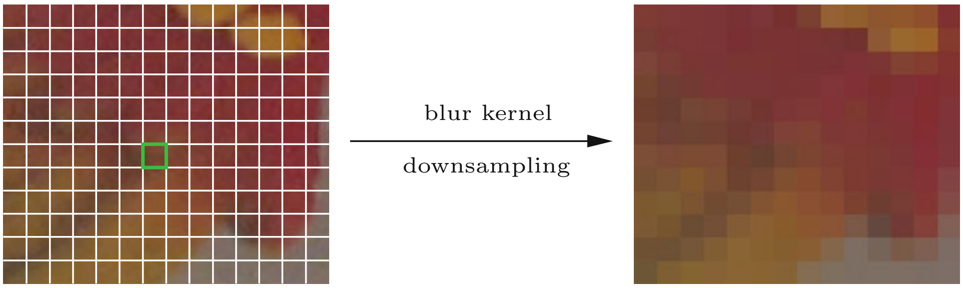


Fig. 3 Super-resolution image formation model. Each sensor element (*green square*) samples incoming light on its surface area. This sampling process is modeled by a convolution kernel b (Sroubek et al. 2007) (Color figure online)

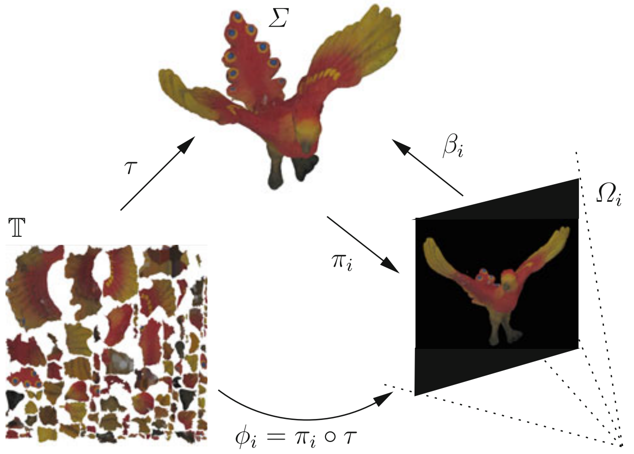


Fig. 4 The various mappings connecting texture space $\mathbb{T}$, the surface Σ and the image planes Ω_i

over the surface. For this, we have to take into account visibility. Let the functions $\mathrm{v}_i : \Sigma \to \{0, 1\}$ indicate whether a surface point is visible in a source image,

$$\mathrm{v}_i(s) := \begin{cases} 1 & \text{if } \pi_i(s) \in S_i \quad \text{and } s = \beta_i \circ \pi_i(s), \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

In addition to the visibility indicator functions, the integral transform requires the area elements $|\det \nabla\pi|$ of the projection map. Note that here and in the following, we use the symbol ∇ not only for the gradient operator, but also to denote the differential (Jacobian matrix) of map which follows it. For reasons of efficiency, it is better to store the area element as an object on each individual image plane, and compute it with respect to the backprojection map β_i. Since π_i and β_i are inverse to each other, we define $\mathcal{J}_i^\pi : \Omega_i \to \mathbb{R}$ via

$$\begin{aligned} \mathcal{J}_i^\pi &:= |\det(\nabla\pi) \circ \beta_i| = |\det(\nabla\beta_i)|^{-1} \\ &= \left| \frac{\partial\beta_i}{\partial x} \times \frac{\partial\beta_i}{\partial y} \right|_2^{-1}. \end{aligned} \tag{4}$$

Here and in the following, $|\cdot|_2$ denotes the usual Euclidean norm. With these definitions, the data term can be transformed as follows. Using the integral substitution formula,

$$E_{\text{data}}(T) = \sum_{i=1}^{n} \int_{\Sigma} \mathrm{v}_i \left| \mathcal{J}_i^\pi \mathcal{E}_i \right| \circ \pi_i \, \mathrm{d}s. \tag{5}$$

Here, the error images $\mathcal{E}_i$ are defined for abbreviation as the difference between the current rendering of the object and the original images,

$$\mathcal{E}_i := b * (T \circ \beta_i) - \mathcal{I}_i. \tag{6}$$

The surface area element $\mathcal{J}_i^\pi$ accounts for foreshortening of the surface in the input views, and is small in regions where the backprojection varies strongly, which is usually the case at silhouette boundaries or discontinuities of the backprojection due to self-occlusions, see Fig. 5. As a consequence, we have the desirable property that in those regions where texture information from the image is unreliable, the input is assigned less weight. Note that we did not need any heuristic assumptions to arrive at this weighting scheme. It is rather a direct consequence of the variational formulation.

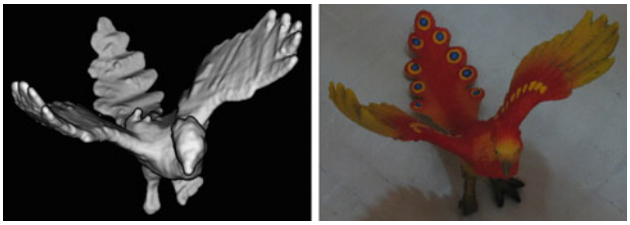


Fig. 5 Intensity-coded backprojection area element $\mathcal{J}_i^\pi$ and corresponding input image. Some errors in the input geometry are clearly visible—no good texture can be expected in those regions

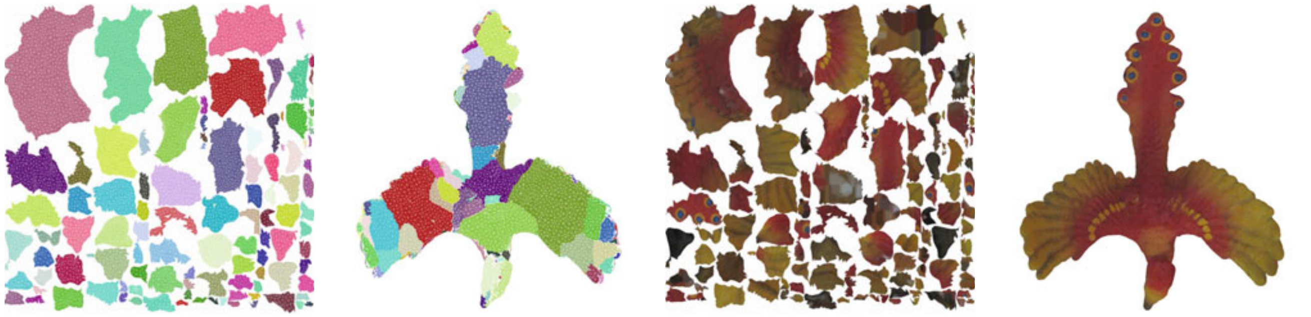
3.3 Surface Parametrization and Conformal Atlas

The energy has now been completely transformed to the surface. In order to be able to minimize it and arrive at a solvable model, however, we need a discretization. In particular, we have to generate a computation grid on which to evaluate the differential operators. While it is possible and has some merits to solve surface PDEs based on an implicit surface representation (Bertalmio et al. 2001; Jin et al. 2008), the goal here are high-resolution textures which exceed the feasible resolution of the voxel grids. Thus, a method based on parametrizing the surface (Lui et al. 2005; Stam 2003) with a texture atlas is preferred for super-resolution.

It turns out that for an elegant mathematical formulation and to achieve the best quality, it is best to opt for a parametrization which is as distortion-free as possible. Such a parametrization is given by a conformal atlas, which will be introduced now.

The idea is to map the texture on the surface to a planar, 2D texture map. In order to achieve this, one needs a global parametrization of the surface. For the purposes here, it is defined as a collection of k maps $\tau_1, \dots, \tau_k$ called the *charts* of the surface. Each chart τ_j maps from an open subset $R_j \subset \mathbb{R}^2$ to an open subset $C_j \subset \Sigma$. We require the following properties to hold:

- The images $C_1, \dots, C_k$ are pairwise disjoint, and the union of their closures $\cup_{j=1}^k \overline{C_j}$ completely covers Σ.
- The different domains $R_1, \dots, R_k$ are also pairwise disjoint.
- Each $\tau_j : R_j \to C_j$ is differentiable and has a continuous inverse, which also implies that it is non-degenerate, i.e. $\det(\nabla\tau_j) \neq 0$.



(a) Chart domains **(b)** Chart images **(c)** Texture map on texture space **(d)** Texture mapped on surface

Fig. 6 Illustration of the charts and parametrization mappings. From *left* to *right* **a** chart domains $\mathrm{dom}(\tau_j)$ in $\mathbb{R}^2$ forming the texture space $\mathbb{T}$. **b** Corresponding regions C_j on the surface. **c** Texture map $\mathcal{T} = T \circ \tau_j$ on texture space. **d** Texture T mapped on surface

The union $\mathbb{T} := \cup_{j=1}^{k} R_j$ of the chart domains is called the texture space. To simplify notation, the single mappings τ_j are combined to form a global mapping $\tau : \mathbb{T} \to \Sigma$. Figure 6 illustrates the concept.

To get optimal texture maps, it is important that the chart mapping τ is as distortion-free as possible, and that the texture is spread evenly over the surface, i.e. the ratio of texture to surface area is similar everywhere. In order to achieve these goals, we demand that τ is *conformal*. Intuitively, this means that it locally preserves angles, i.e. is locally essentially the same as a rotation and scaling operation—mathematically speaking, the Jacobian matrix $\nabla\tau$ can be written as a rotation matrix times an isotropic scaling by a factor $c > 0$. This factor is called the *conformal factor* of the mapping τ, and is identical to the area element $\det(\nabla\tau)$.

To implement a general surface parametrization, numerous methods are available (Floater and Hormann 2006). However, when one needs a conformal parametrization, the options are more limited (Gu and Yau 2003; Lévy et al. 2003; Wang et al. 2005). In this paper, we use a straightforward implementation of Lévy et al. (2003). As already explained above, an even coverage of the model with texels is desireable, so charts are split up if the ratio of largest to smallest conformal factor is greater than 3/2. Chart domains are then scaled so that the average conformal factor equals a common constant. The algorithm is fully automatic and has the desirable property that chart boundaries tend to coincide with high-curvature edges on the surface. The input mesh needs to be a defect-free manifold mesh for a closed 2D surface.

3.4 Final Model on Texture Space

Using the conformal atlas, we transform the energy to texture space and arrive at the final discretized model which we are going to solve. Let $c : \mathbb{T} \to \mathbb{R}$ be the mapping assigning the conformal factor to each point in texture space. We first transform the regularizer. Note that the total variation of the texture map can be computed on texture space by pulling back the integral,

$$\mathrm{TV}(T) = \int_{\Sigma} \|\nabla_{\Sigma} T\|_{\Sigma} \; \mathrm{d}s = \int_{\mathbb{T}} \sqrt{c}\, |\nabla \mathcal{T}|_2 \; \mathrm{d}x. \tag{7}$$

See Lui et al. (2005) for a detailed explanation of the transformation formulas. Intuitively, a conformal map scales tangent vectors uniformly, so lengths scale with the square root of the area element, i.e. the conformal factor.

The data term is simple to transform, since the conformal factor is equal to the surface area element of the transformation τ. The concatenation $\phi_i := \pi_i \circ \tau$ yields the map from texture space to input image for each camera, see Fig. 4. We obtain

$$\begin{aligned} E_{\text{data}}(T) &= \sum_{i=1}^{n} \int_{\Sigma} \mathrm{v}_i \left| \mathcal{J}_i^{\pi} \mathcal{E}_i \right| \circ \pi_i \; \mathrm{d}s \\ &= \sum_{i=1}^{n} \int_{\mathbb{T}} \left(\mathrm{v}_i \left| \mathcal{J}_i^{\pi} \mathcal{E}_i \right| \circ \phi_i \right) c \; \mathrm{d}x. \end{aligned} \tag{8}$$

For abbreviation, we set $\mathcal{J}_i^{\phi} := (c \circ \phi_i^{-1}) \mathcal{J}_i^{\pi}$. Note for later that $\mathcal{J}_i^{\phi}$ is thus the surface area element with respect to the transformation ϕ_i. In summary, the energy on texture space in terms of $\mathcal{T}$ is now given by

$$E(\mathcal{T}) = \int_{\mathbb{T}} \lambda\sqrt{c}\, |\nabla \mathcal{T}|_2 + \sum_{i=1}^{n} \mathrm{v}_i \left| \mathcal{J}_i^{\phi} \mathcal{E}_i \right| \circ \phi_i \; \mathrm{d}x, \tag{9}$$

with $\mathcal{E}_i$ defined in Eq. (6).

3.5 Discretization

For discretization, the texture space is subdivided into a grid of texels. The grid needs to admit a flexible topology, since only texels in the interior of chart domains are connected to their direct neighbours. Note that the surface is closed, and technically we have a form of periodic boundary conditions.

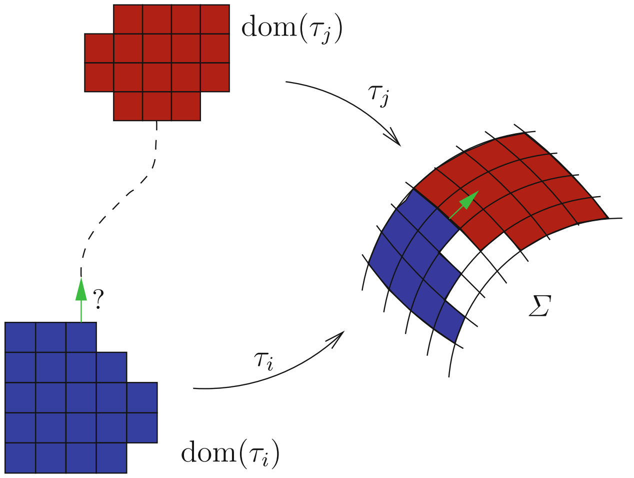


Fig. 7 Boundary texel neighbour connections on the computation grid are established by searching in normal direction on the surface

However, because a global parametrization usually requires several charts, $\mathbb{T}$ is made up of several disjoint domains. On the boundary of charts, neighbourhood is established according to the surface topology. To achieve this, we take the outer normals of boundary texels in texture space and transform them up onto the surface. Then, we search for the closest texel of the neighboring chart in this direction, and assign that one as a neighbour, see Fig. 7.

After having established boundary transitions, the gradient ∇ is then discretized with standard forward differences. The divergence is computed as the negative transpose of the gradient, div $:= -\nabla^T$, since the discrete version of the theorem of Gauss has to be satisfied on the surface.

4 Optimizing for the Texture Map

The final model consists of a convex regularizer with convex data term on the flat 2D texture space. We have explained the discretization of the operators on texture space, so we have arrived at a state where we can apply a standard primal–dual minimization scheme (Chambolle and Pock 2011). However, since the implementation of the super-resolution model is very involved, we will give some more details and a description of the necessary implementation steps in this section.

4.1 Setting Up the Computations

The first step is to construct a conformal atlas as described in Sect. 3.3, which yields the computation grid on which we discretize and solve the model. On the grid, we set up the matrix for the gradient operator using simple forward differences, and its negative transpose, which is the divergence according to the Gaussian theorem. From the atlas, we obtain boundary conditions for these operators using the neighbourhood relationships of the charts, as described in Sect. 3.5.

In order to pre-compute $\mathcal{J}_i^\pi$ for each camera, we need to obtain the backprojection mapping β_i. An efficient way to do this is by rendering the surface into each camera (e.g. using OpenGL) and then reading the Z-buffer to get the exact location for the backprojection of each pixel. From this information, we can also extract the visibility indicator function v_i. The area element $\mathcal{J}_i^\pi$ is then obtained numerically via Eq. (4) using central differences. Similarly, we can compute the conformal factor c for the atlas, while $\mathcal{J}_i^\phi$ is just the product of the two. We then have all pre-requisites information in place to minimize energy (9) with respect to the texture map.

4.2 Saddle Point Formulation of the Problem

The model (9) has both a non-differentiable regularizer as well as data term. While it is by now common knowledge how to perform total variation minimization with arbitrary convex data terms (Chambolle and Pock 2011), the L^1 data term is slightly more involved in this model due to the different domain transformations. Using the second convex conjugate or biconjugate for the L^1-norm, see Sect. 9.3 in Attouch et al. (2006), we first note that

$$\begin{aligned} E^i_{\text{data}}(\mathcal{T}) &= \int_{\mathbb{T}} \mathrm{v}_i \left| \mathcal{J}_i^\phi \mathcal{E}_i \right| \circ \phi_i \, \mathrm{d}x \\ &= \max_{|q_i| \le \mathrm{v}_i} \left(q_i, \left(\mathcal{J}_i^\phi \mathcal{E}_i \right) \circ \phi_i \right), \end{aligned} \tag{10}$$

where $(u, v) = \int_{\mathbb{T}} uv \, \mathrm{d}x$ denotes the inner product on the Hilbert space $L^2(\mathbb{T})$. The maximum is computed over a vector $\boldsymbol{q} = (q_i)_{i=1,\ldots,n}$ of scalar fields on $\mathbb{T}$, one for each camera, whose absolute value is point-wise less or equal to v_i. Together with the well-known dual formulation for the weighted total variation (Bresson et al. 2007), one can transform the inverse texture reconstruction problem (9) into the saddle point problem

$$\min_{\mathcal{T}} \max_{\substack{|q_i| \le \mathrm{v}_i \\ |\boldsymbol{\zeta}|_2 \le \lambda\sqrt{c}}} E(\mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{q}), \tag{11}$$

with the primal–dual energy

$$E(\mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{q}) := -(\mathcal{T}, \operatorname{div}(\boldsymbol{\zeta})) + \sum_i \left(q_i, \left(\mathcal{J}_i^\phi \mathcal{E}_i \right) \circ \phi_i \right). \tag{12}$$

In this form, the problem is now ready to be implemented.

4.3 Saddle Point Solver for the Primal–Dual Energy

In Chambolle and Pock (2011), Pock et al. (2009), a generic algorithm is given how to solve saddle point problems of type (11). We implement their method in this work, which requires computation of all the derivatives of $E(\mathcal{T}, \boldsymbol{\zeta}, q)$ with respect to the primal variable $\mathcal{T}$ and dual variables $\boldsymbol{\zeta}$, q, respectively.

The derivative with respect to q_i is simple, since E is linear in this variable, and given by

$$\begin{aligned}\frac{\partial E}{\partial q_i}(\mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{q}) &= \left(\mathcal{J}_i^{\phi} \mathcal{E}_i\right) \circ \phi_i \\ &= \left(\mathcal{J}_i^{\phi} \left(b * \left(\mathcal{T} \circ \phi_i^{-1}\right)\right) - \mathcal{I}_i\right) \circ \phi_i. \end{aligned} \tag{13}$$

By construction of the operators, $-\mathrm{div} = \nabla^T$, thus the derivative with respect to $\boldsymbol{\zeta}$ also does not pose a problem,

$$\frac{\partial E}{\partial \boldsymbol{\zeta}}(\mathcal{T}, \boldsymbol{\zeta}, qv) = \nabla \mathcal{T}. \tag{14}$$

More thought has to be put into the derivative with respect to $\mathcal{T}$. One can transform the inner product in the data term as

$$\begin{aligned}\left(q_i, \left(\mathcal{J}_i^{\phi} \mathcal{E}_i\right) \circ \phi_i\right) &= \left(q_i \circ \phi_i^{-1}, \mathcal{E}_i\right) \\ &= \left(q_i \circ \phi_i^{-1}, b * \left(\mathcal{T} \circ \phi_i^{-1}\right) - \mathcal{I}_i\right) \\ &= \left(\bar{b} * \left(q_i \circ \phi_i^{-1}\right), \mathcal{T} \circ \phi_i^{-1}\right) \\ &\quad - \left(q_i \circ \phi_i^{-1}, \mathcal{I}_i\right) \\ &= \left(\mathcal{J}_i^{\phi} \left(\bar{b} * \left(q_i \circ \phi_i^{-1}\right)\right) \circ \phi_i, \mathcal{T}\right) \\ &\quad - \left(q_i \circ \phi_i^{-1}, \mathcal{I}_i\right), \end{aligned} \tag{15}$$

with the transposed kernel $\bar{b}(x) := b(-x)$. Note that the area element $\mathcal{J}_i^{\phi}$ with respect to ϕ_i disappears in line two and appears back later because one performs integral transformations between texture space and image space. From (15), one can conclude

$$\begin{aligned}\frac{\partial E}{\partial \mathcal{T}}(\mathcal{T}, \boldsymbol{\zeta}, \boldsymbol{q}) &= -\mathrm{div}(\boldsymbol{\zeta}) \\ &\quad + \sum_{i=1}^{n} \left(\mathcal{J}_i^{\phi} \left(\bar{b} * \left(q_i \circ \phi_i^{-1}\right)\right)\right) \circ \phi_i. \end{aligned} \tag{16}$$

For efficient computation of the derivative with respect to q_i in Eq. (13), remember that $\mathcal{T} \circ \phi_i^{-1}$ is simply a rendering of the surface into the ith image using the current texture map and a pinhole camera model, which can efficiently be performed with OpenGL. A high-resolution rendering is required, which is then convolved with b and subsampled to the resolution of the input images, following the imaging model of the camera. Computing the difference to $\mathcal{I}_i$ does not pose a problem, and concatenation with ϕ_i corresponds to a lookup for the texel at the corresponding location in the target image, which is simply a forward projection which can be computed on the fly.

In a similar way, one can proceed with the terms in the derivative with respect to $\mathcal{T}$, using the dual variable q_i in place of the texture map during rendering. All those operations can be implemented on the GPU for significant speedup.

The complete algorithm to estimate a super-resolved texture, which resembles the first algorithm in Chambolle and Pock (2011) but is directly adapted to our case, can be found in Fig. 8.

4.4 Extension to Color Textures

Extending our model to color textures just requires an appropriate multidimensional total variation norm in the energy functional. In Goldluecke et al. (2012), several choices are presented, together with the straight-forward generalization

Texture super-resolution

Initialize

fields on $\mathbb{T}$:

textures $\mathcal{T}_0 = 0$, $\bar{\mathcal{T}}_0 = 0$

a vector $\boldsymbol{q}_0 = 0$ of n scalar fields

a vector field $\boldsymbol{\zeta}_0 = 0$

real numbers:

step sizes $\tau = \sigma = \frac{1}{8}$

overrelaxation factor $\theta = 1$

Iterate

dual variable update

$$\boldsymbol{\zeta}_{n+1} = \Pi_{\lambda\sqrt{c}}(\boldsymbol{\zeta}_n + \sigma \nabla \bar{\mathcal{T}}_n)$$

$$\boldsymbol{q}_{n+1} = \Pi_{\boldsymbol{v}}\left(\boldsymbol{q}_n + \sigma \frac{\partial E}{\partial \boldsymbol{q}}(\bar{\mathcal{T}}_n, \boldsymbol{\zeta}_n, \boldsymbol{q}_n)\right)$$

primal variable update

$$\mathcal{T}_{n+1} = \mathcal{T}_n - \tau \frac{\partial E}{\partial \mathcal{T}}(\mathcal{T}_n, \boldsymbol{\zeta}_{n+1}, \boldsymbol{q}_{n+1})$$

extragradient step

$$\bar{\mathcal{T}}_{n+1} = \mathcal{T}_{n+1} + \theta(\mathcal{T}_{n+1} - \mathcal{T}_n)$$

with projections

$$\Pi_{\lambda\sqrt{c}}(\boldsymbol{\zeta}) = \frac{\lambda\sqrt{c}\,\boldsymbol{\zeta}}{\max\left(\lambda\sqrt{c}, |\tilde{\boldsymbol{\zeta}}|_2\right)}$$

$$\Pi_{\boldsymbol{v}}(q_i) = \frac{v_i q_i}{\max(v_i, |q_i|)}$$

and derivatives

$\frac{\partial E}{\partial \boldsymbol{q}}$ given component-wise in (13)

$\frac{\partial E}{\partial \mathcal{T}}$ given in (16)

Fig. 8 Texture super-resolution algorithm which finds a saddle point of energy (12). After convergence, the last iterate $\mathcal{T}_n$ is the solution of the problem. The above method is a specialization of Algorithm 1 in Chambolle and Pock (2011), and essentially performs gradient ascent and reprojection for the dual variables and gradient descent and reprojection for the primal variables, respectively. The extragradient step leads to an accelerated convergence. The operators Π_r denote the pointwise projection onto a ball of radius r of suitable dimension, as defined above

of the algorithm above. When implementing it in practice, we just need to initialize separate variables for each channel and perform all steps of the iterations independently, except for the reprojection for the dual variables $\boldsymbol{\zeta}$, which is joined for all channels. For details, we refer to Goldluecke et al. (2012).

5 Optimizing for a Surface Displacement Map

In the optimization step for the displacement map, we assume that we have an estimate for the object texture available. Using this estimate, we locally improve the geometry to yield a better fit of the rendered model to the input images.

5.1 Displacement Map Model

The unknown in this step is a *displacement map* $D : \Sigma \to \mathbb{R}$, which assigns to each point on the surface a displacement quantity measured as a multiple of the outer unit normal $\mathbf{n}$ of the surface. We compute the solution on texture space as well, where the unknown displacement map transforms to $\mathcal{D} := D \circ \tau$. Note that the forward projection maps $\phi_i : \mathbb{T} \to \Omega_i$, and in consequence the difference images $\mathcal{E}_i$, now depend on the displacement, see Fig. 9. We clarify this by adding a superscript, and write $\phi_i^{\mathcal{D}}$ and $\mathcal{E}_i^{\mathcal{D}}$ instead.

The super-resolution functional (9) on texture space is now extended to include the model for the unknown displacement map $\mathcal{D}$, which we also regularize with the total variation,

$$E(\mathcal{T}, \mathcal{D}) = \int_{\mathbb{T}} \lambda\sqrt{c}\,|\nabla\mathcal{T}|_2 + \mu\sqrt{c}\,|\nabla\mathcal{D}|_2 \\ + \sum_{i=1}^{n} \mathrm{v}_i \left|\mathcal{J}_i^{\phi}\mathcal{E}_i^{\mathcal{D}}\right| \circ \phi_i^{\mathcal{D}}\,\mathrm{d}x. \qquad (17)$$

The parameter $\mu > 0$ adjusts the smoothness of the displacement map.

We omit the dependency on the texture map in the following for clarity of notation, and focus only on displacement optimization. Later on, we jointly optimize the functional for both unknowns by means of alternating optimization in $\mathcal{T}$ and $\mathcal{D}$.

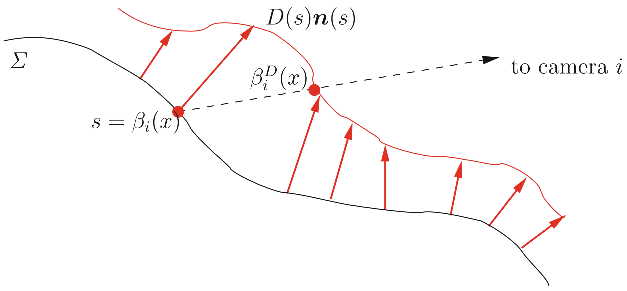


Fig. 9 The displacement map D assigns to each point on the surface a shift in normal direction, which leads to non-local changes in the backprojection

Note that in contrast to texture optimization, the data term is non-convex in the displacement map $\mathcal{D}$. Since local optimization via gradient descent is prone to end up in a bad local minimum, we therefore need a different approach to minimization. For simplification, we need to assume that for each point, $\mathcal{D}$ is constant in the sampling area of the kernel b. In this case, the data term is defined point-wise, and the energy takes the form

$$E(\mathcal{D}) = \int_{\mathbb{T}} \mu\sqrt{c}\,|\nabla\mathcal{D}(x)|_2 + \rho(\mathcal{D}(x),\, x)\,\mathrm{d}x, \qquad (18)$$

where $\rho : \mathbb{R} \times \mathbb{T} \to \mathbb{R}$ is given by the corresponding term in Eq. (17), i.e.

$$\rho(\mathcal{D}(x),\, x) := \sum_{i=1}^{n} \mathrm{v}_i(x) \left|\mathcal{J}_i^{\phi}(x)\mathcal{E}_i^{\mathcal{D}(x)}(x)\right| \circ \phi_i^{\mathcal{D}(x)}(x). \qquad (19)$$

5.2 Optimization by Quadratic Relaxation

In order to find a local minimizer of the energy, we propose the method of quadratic relaxation. It was successfully employed in previous work for the optical flow functional (Zach et al. 2007), a method we also use again in later sections. While in theory it would be possible to perform a global minimization of the functional based on the functional lifting idea (Pock et al. 2008), this would be prohibitively slow in this setting and also only allow a small set of discrete displacement labels due to memory limitations.

We introduce an auxiliary variable $\mathcal{U}$ and decouple the regularization term from the point-wise optimization by defining an approximation to the energy in two variables,

$$E(\mathcal{D}, \mathcal{U}) = \int_{\mathbb{T}} \mu\sqrt{c}\,|\nabla\mathcal{U}(x)|_2 + \frac{1}{2\theta}(\mathcal{U} - \mathcal{D})^2 \\ + \rho(\mathcal{D}(x),\, x)\,\mathrm{d}x. \qquad (20)$$

For $\theta \to 0$, the solution of this auxiliary problem approaches the solution to the original problem, as the coupling term forces $\mathcal{U}$ to be close to $\mathcal{D}$. The idea is that for fixed $\mathcal{U}$, we can perform a point-wise optimization in $\mathcal{D}$, since no spatial derivatives of $\mathcal{D}$ appear in the functional. This can be done in a globally optimal way by means of an exhaustive search in the range of allowed displacement values. On the other hand, for fixed $\mathcal{D}$, the resulting energy functional resembles the ROF denoising model (Rudin et al. 1992), which is convex and thus can also be optimized globally (Chambolle and Pock 2011; Goldluecke et al. 2012). Thus, by alternating two

Displacement map optimization

Initialize scalar fields $\mathcal{D} = \mathcal{U} = 0$ on texture space. Then alternate between updating $\mathcal{U}$ and $\mathcal{D}$ by solving the following two optimization problems until convergence.

- For $\mathcal{D}$ fixed, find the minimizer $\mathcal{U}$ for the ROF part of energy (20),

$$\int_{\mathbb{T}} \mu\sqrt{c}\,|\nabla\mathcal{U}(x)|_2 + \frac{1}{2\theta}(\mathcal{U}-\mathcal{D})^2\,\mathrm{d}x, \tag{21}$$

  using any of the methods in [18], with projections adjusted for weighted total variation [8].
- For $\mathcal{U}$ fixed and every $x \in \mathbb{T}$, find the global optimum $\mathcal{D}(x)$ of the point-wise term in (20),

$$\frac{1}{2\theta}(\mathcal{U}(x)-\mathcal{D}(x))^2 + \rho(\mathcal{D}(x), x) \tag{22}$$

  using a brute-force search in the allowed displacement range.

Fig. 10 Algorithm to optimize for the displacement map, given the surface texture and camera calibration

global optimization steps, one can arrive at a good minimizer for the original energy (17), which will however in the general case be only a local minimum.

The complete algorithm is summarized in Fig. 10. Note that in order to compute ρ, one has to render the current model using a displacement map. For this, we employ an OpenGL implementation of the algorithm described in Donnelly (2005).

6 Optimizing for the Projection Parameters

6.1 Variational Camera Calibration Model

Given a texture and the 3D model, the optimal camera parameters are those which minimize the reprojection error. In this section, we assume that we have a pre-computed super-resolved texture map $\mathcal{T}$ and surface geometry Σ, possibly adjusted by a displacement map. The goal is to minimize the energy (1) with respect to the projection parameters π_i in order to obtain a more accurate calibration. Of course, once the calibration becomes more accurate, both the 3D model as well as the texture map can then iteratively be improved using the optimization methods in the previous sections.

The derivatives of the back-projection β are very difficult to compute and depend on the 3D model, whose accuracy is hard to predict. For this reason, we choose version (5) of the data term on the surface as an initial model. Slightly rewritten and simplified, we obtain

$$E_{\text{data}}(\mathcal{T}, \mathcal{D}, \boldsymbol{\pi}) = \sum_{i=1}^{n} \int_{\Sigma} \mathrm{v}_i \left(\mathcal{J}_i^{\pi} \circ \pi_i\right) |T - \mathcal{I}_i \circ \pi_i|\, \mathrm{d}s. \tag{23}$$

as the total reprojection error E depending on the set of all projection parameters $\boldsymbol{\pi}$ as well as texture and displacement map. After discretization, this energy is simply computed as a sum over all the vertices of the surface mesh, weighted with the projection errors, visibility indicator functions v_i and surface area elements $\mathcal{J}_i^{\pi}$. As before, we omit the dependency on texture map and displacement map in the following for clarity of notation, and to focus on the optimization of projection parameters.

Note that we need to make a few simplifications to make optimization computationally viable. First, we exploit the super-resolution model only for computing an accurate texture map, since it would be computationally prohibitive to optimize the full model with respect to the projections. Therefore, to obtain (23), the convolution with the kernel b is set to identity (which corresponds to a Dirac kernel). Second, the visibility v_i and surface area element $\mathcal{J}_i^{\pi}$ are costly to compute, and kept constant during optimization, assuming that they are relatively stable under small changes of the projection.

6.2 Direct Minimization via Gradient Descent

Note that if T is kept fixed, each term of the sum in (23) is completely independent of the others, and can be minimized separately. For this reason, we will consider in the remainder of the section only a single camera, and omit the dependency on the index i to simplify notations.

In order to minimize the respective term in energy (23) with respect to π, we need a suitable parametrization of π by a set of parameters $(g_j)_{1\leq j\leq m}$. In our implementation, we limit ourselves to the camera extrinsics, i.e. rotation and translation, giving rise to $(m=6)$ degrees of freedom. Yet, a generalization to a more complex camera model is straightforward.

The most simple approach to minimization is to perform gradient descent in the parameters g_i. Since the energy is nonlinear, we iterate a sequence of update steps, where in each step we minimize the linearized energy

$$\int_{\Sigma} \mathrm{v}\left(\mathcal{J}^{\pi} \circ \pi\right) \left| \mathcal{I} \circ \pi - T + \nabla\mathcal{I}\left(\sum_{j=1}^{m} \frac{\mathrm{d}\pi}{\mathrm{d}g_j}\delta g_j\right)\right| \mathrm{d}s, \tag{24}$$

with respect to calibration parameter updates $\delta g_1, \ldots, \delta g_m$. In each step, this is a standard least-squares problem, which can easily be solved with off-the-shelf algorithms. To compute the derivatives of π, we make use of the exponential parametrization. For technical details, we refer to Unal et al. (2007).

The above gradient descent approach is a slight generalization of the model in Unal et al. (2007). While also relying on the reprojection error, it differs in two significant ways. Firstly, it relies on a super-resolution formulation which is expected to capture finer object texture details and thus leads to a more precise calibration. Secondly, we exchange the L^2-norm used in Unal et al. (2007) with the L^1-norm which is theoretically more robust and which we found to give better results in practice. Furthermore, experiments indicate that local minimization of this highly non-convex optimization problem gives rise to suboptimal solutions and often does not lead to substantial improvements in the estimated camera parameters. In the following, we will provide reasons for this shortcoming and propose a decoupling strategy which leads to a considerably more robust calibration method.

6.3 Decoupling Photometric and Geometric Error

A closer look at functional (23) reveals that the camera parameters are being estimated so as to minimize the *photometric* error between modeled and observed texture. Interestingly, this is in sharp contrast to the established bundle adjustment approach for accurate camera calibration which aims at minimizing the *geometric* error between observed points and the corresponding back-projected 3D points. In particular, the problems of estimating point correspondences and minimizing the geometric error are treated separately. This is important because, upon gradual improvement of the camera parameters, the geometric error is likely to decrease, whereas — in particular for high-resolution textures—the photometric error is more likely to oscillate (rather than decrease), thus leading to bad convergence of algorithms based on pure photometric criteria. In other words, incorporating a geometric measure reduces the number of local minima and gives clearer evolution directions. Furthermore, the success of established tools like bundle adjustment indicates that for accurate camera calibration based on high-resolution textures, one should separate the algorithmic problems of correspondence estimation and calibration.

6.4 Decoupled Energies and Optimization

The central idea is to first explain the photometric error $\left|\mathcal{I} - \mathcal{T} \circ \phi^{-1}\right|$ for each view by means of a smooth *dense displacement field* $v : \Omega \to \mathbb{R}^2$, which minimizes the photometric error $\left|\mathcal{I}(\mathrm{id}_\Omega + v) - \mathcal{T} \circ \phi^{-1}\right|$ between warped source image $\mathcal{I}(\mathrm{id}_\Omega + v)$ and rendering $\mathcal{T} \circ \phi^{-1}$ of the object in the image plane, where id_Ω denotes the identity map on Ω (Fig. 11). Only afterwards, we account for this error by optimizing the projections. The field v can be interpreted as the *geometric reprojection error* for the current set of projection parameters. In effect, we thus decouple minimization of the photometric and the geometric error.

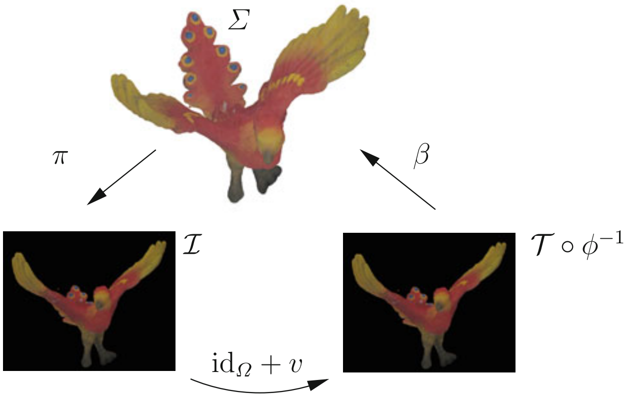


Fig. 11 The image $\mathcal{T} \circ \phi^{-1}$ is obtained by rendering the current textured 3D model by means of the back-projection map β of each camera. We propose to use the optic flow v between $\mathcal{I}$ and $\mathcal{T}$ as a measure for the geometric reprojection error and minimize it with respect to the projection π

We first show how to estimate the reprojection error v in the image plane by minimizing the photometric error. We regularize v with the vectorial total variation, and obtain an estimate by minimizing the $TV - L^1$ functional (Zach et al. 2007)

$$E_1(v) = \int_S \left|\mathcal{I}\left(\mathrm{id}_\Omega + v\right) - \mathcal{T} \circ \phi^{-1}\right| + \alpha \left\|Dv\right\|_F \,\mathrm{d}x, \tag{25}$$

which is defined inside the silhouette $S \subset \Omega$ of the object. The parameter $\alpha > 0$ controls the desired smoothness of the result.

In the second step, we show how to optimize the projections π by accounting for the estimated reprojection error v. We need to determine new camera parameters which account for the displacement v. For this, we update the current camera parameters $\tilde{\pi}$ by minimizing

$$E_2(\pi) = \int_\Sigma \mathrm{v}\left(\mathcal{J}^\pi \circ \pi\right) \left|\pi - \left(\mathrm{id}_\Omega + v\right) \circ \tilde{\pi}\right| \,\mathrm{d}s. \tag{26}$$

The idea is that the updated projected locations should be close to the old locations displaced by the vector field v. Equation (26) resembles the standard discrete bundle adjustment process in which $\sum_j (x_j - \pi(s_j))^2$ is being minimized, where s_j are the different discrete 3D points and x_j the corresponding projections. Note that we have an additional term which accounts for the foreshortening of the surface under perspective projection. In fact, modeling this important aspect in a rigorous way is only possible in a con-

```
Variational camera calibration

Iterate for all cameras $i = 1, \dots, n$ until convergence:
  – compute $T \circ \beta_i$
  – compute optical flow $v$ between $\mathcal{T} \circ \phi_i$ and $\mathcal{I}_i$ by minimizing $E_1(v)$
  – compute $\{\delta g_j\}_{j=1,\dots,m}$ to minimize
        $\int_{\Sigma} v_i(\mathcal{J}_i^{\pi} \circ \pi_i) \left| \sum_{j=1}^{m} \frac{\mathrm{d}\pi_i}{\mathrm{d}g_j} \delta g_j - v \circ \pi_i \right| \mathrm{d}s.$
  – Update projection parameters for all $j = 1, \dots, m$ via
        $g_j \leftarrow g_j + \delta g_j .$
  – Construct new $\pi_i$ from updated projection parameters $\{g_j\}_{j=1,\dots,m}$ .
```

Fig. 12 Algorithm for decoupled variational calibration, given estimated surface texture and geometry

tinuous formulation. We minimize E_2 via gradient descent using a linearization of π with respect to the parameter updates $(\delta g_1, \dots, \delta g_m)$, as already described for the energy in Sect. 6.2. The resulting algorithm to optimize for the projection parameters is summarized in Fig. 12.

Comparing the energy functionals in Eqs. (25) and (26) to the methodology of sparse calibration methods, we observe that the proposed dense formulation allows to propagate neighboring information by regularizing the underlying displacement field (v). Thereby, well-textured regions prevail, while homogeneous regions give rise to displacements close to zero. It should be noted that the integral in (25) can easily be "sparsified" by using a weighting function $w : \Omega \to \{0, 1\}$ (or a relaxed version $w : \Omega \to [0, 1]$) which accounts for the reliability of the respective pixel measurement. In practice, however, we found that this is not necessary.

7 Results

We evaluate the super-resolution framework on a number of different synthetic and real-world data sets in order to assess performance of the individual components.

7.1 Parameter Settings and Computation Time

A difficult choice for our algorithm is the convolution kernel b modeling the sensor elements. While it would be optimal to measure it individually for each image acquisition system, we did not have the hardware available. Following the advice in Baker and Kanade (2002), we chose a Gauss kernel with standard deviation equal to half the pixel diameter as a good general approximation.

As we aim at crisp texture maps, the smoothness parameter λ for the texture map should be as low as possible to still produce stable results. In our experiments, we kept it at $\lambda = 0.01$. The smoothness factor μ for the displacement map needs to be larger to suppress outliers, for our data sets $\mu = 0.5$ led to satisfactory results. The optimal smoothness factor α for the optical flow during camera calibration depends on the data set, values around $\alpha = 0.1$ are usually a good choice.

The algorithm was tested on a 2.8 GHz Core 2 Duo processor with CUDA enhancements running on a GeForce GTX 280. Main memory required is around 6 GByte. Initial computation of the texture atlas is not parallelized and takes around 10 min. After that, computation of a super-resolved texture map runs on the GPU and a complete run of the algorithm in Fig. 8 only requires around half a minute.

The optimization for the displacement map runs partly on the GPU but is much more expensive, since it requires recomputation of the backprojection mappings in each iteration. A run of the algorithm in Fig. 10 takes around 10 min until convergence, but we need to iterate ten times with recomputation of the super-resolved texture until we have a final result.

Variational camera calibration performs comparatively fast. Its run-time scales linearly with the number of cameras, and it can be included into the iteration process above before optimizing for the displacement map at the cost of a few seconds per camera. In total, the run-time to compute displacement map, variational calibration as well as a super-resolved texture from the given model and input images is about 2 h.

7.2 Texture Reconstruction from Synthetic Data

In order to evaluate the algorithm numerically and to make meaningful comparisons of texture estimation errors, we first run it on synthetic test scenes. For complete knowledge of all mappings and easy comparison with ground truth we chose a torus as a test object, with a single texture wrapped around it using the standard parametrization. Images of the torus were raytraced from 48 camera locations, distributed evenly around it on six different height levels. Image resolution is 512 × 512 pixels, while the original and reconstructed texture is of size 1, 024 × 1, 024 texels.

For initialization, each texel is set to the weighted average color of its projections in all views where it is visible. Weights are given by the inverse backprojection area element, see Eq. (4). As can be seen in Fig. 13, this common pointwise approach to texture estimation leads to slightly blurry results in which many of the original details are lost. Naturally, the results will be even worse if the geometry is less exact than in this ideal example.

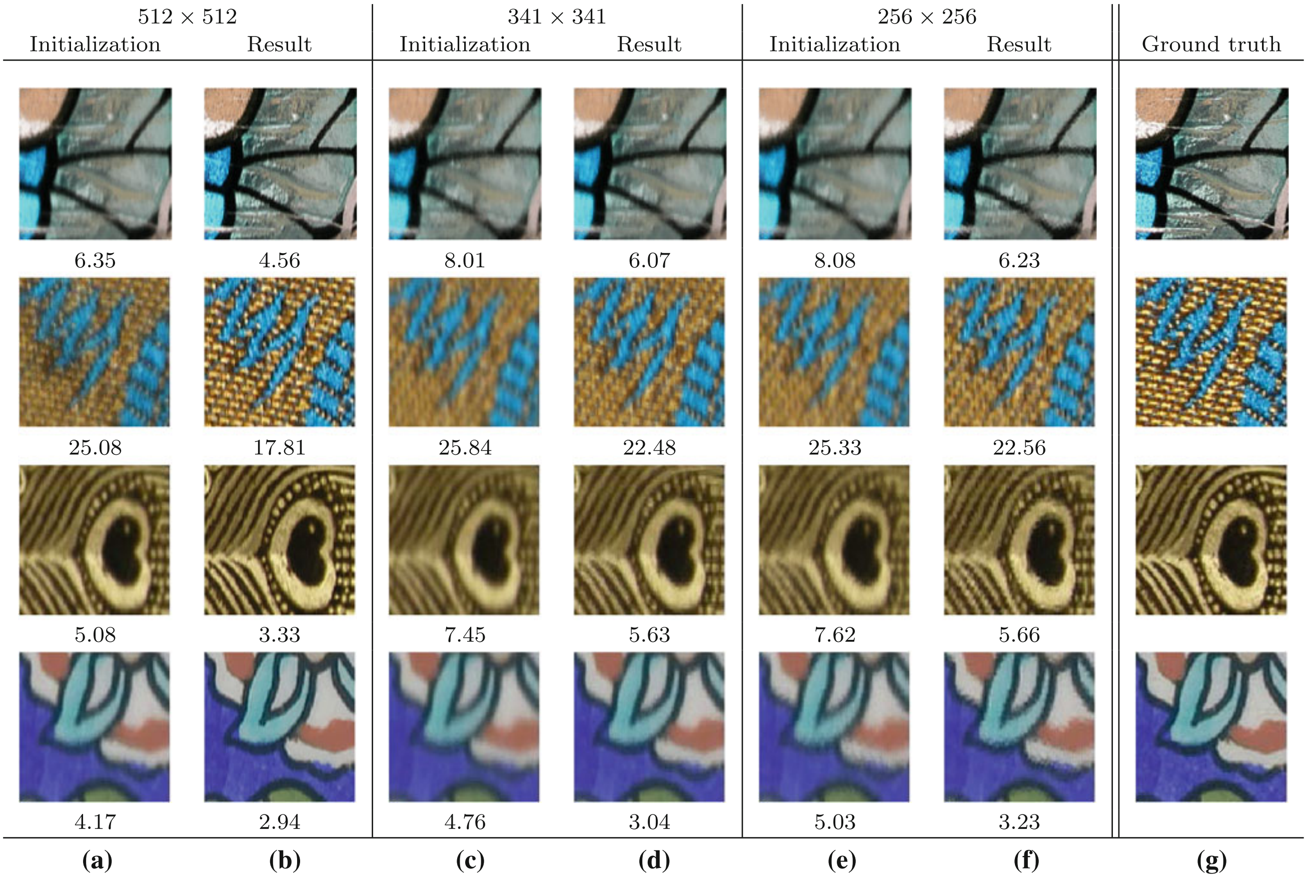
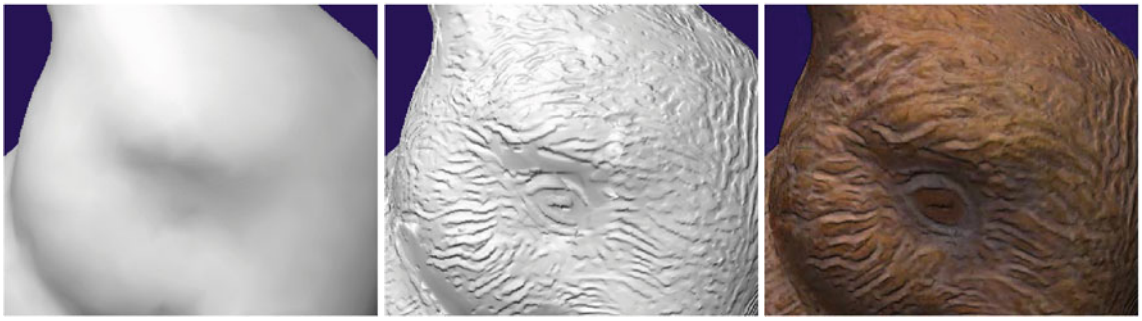


Fig. 13 In the table above, we compare texture initializations computed as the point-wise average (a,c,e) with the super-resolution results (b,d,f) at three different input image resolutions. Super-resolution results from full resolution input images (**b**) are visually almost indistinguishable from ground truth (**g**). Furthermore, super-resolution reconstructions from downscaled images (**f**) are on average of better quality than the texel-wise initialization from four times the input image resolution (**a**). Mean squared error ϵ for a color range of [0, 255] is shown below the image close-ups

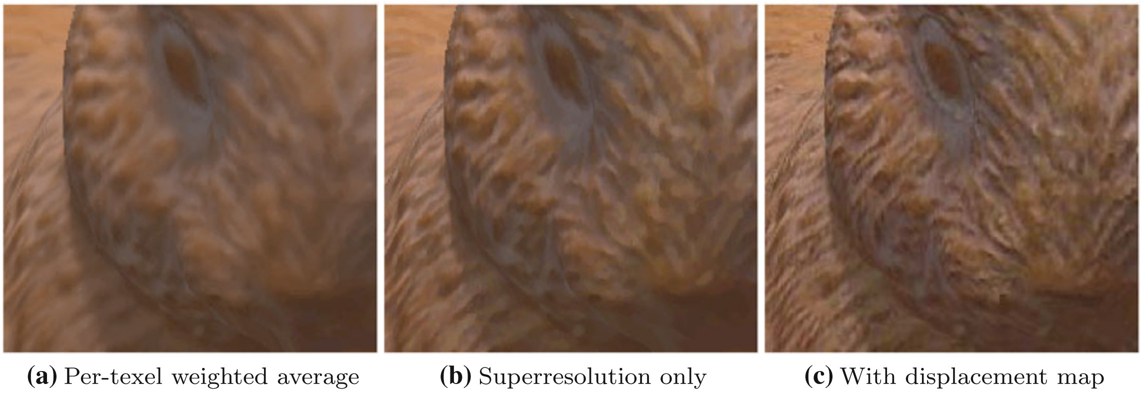


(a) Input mesh **(b)** Estimated displacement map **(c)** With super-resolved texture

Fig. 14 Estimated displacement map for the *Bunny* dataset. From *left* to *right* **a** rendering with Gouraud shading. The underlying mesh has low geometric detail. **b** Normal map lighting showing improved geometric detail from the estimated displacement map. **c** Rendering with super-resolution texture and normal map lighting

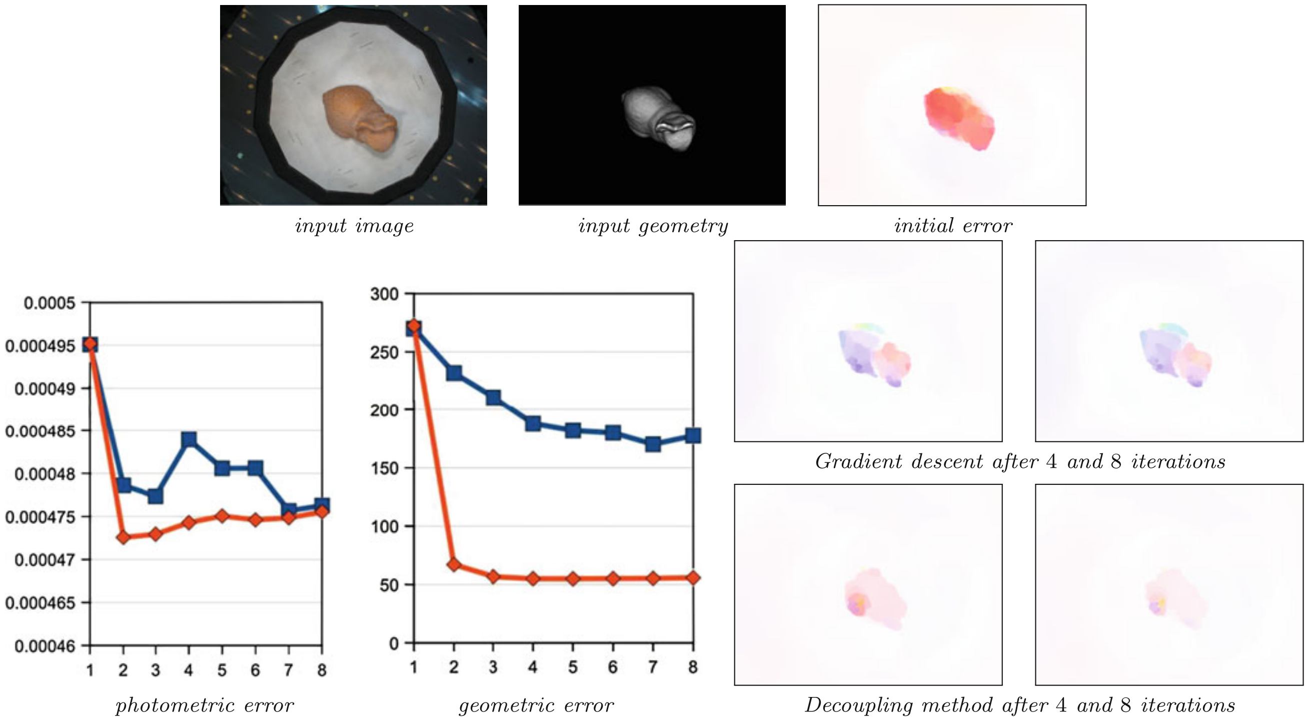
To investigate how the result depends on input image resolution, the super-resolution algorithm is applied to downsampled as well as the original images. In each case, the mean squared error ϵ between reconstructed and original texture map was recorded. Error values and close-ups of the resulting texture maps are shown in Fig. 13.

If the input images are scaled down by a factor of two, our method still produces textures which are comparable to



(a) Per-texel weighted average **(b)** Superresolution only **(c)** With displacement map

Fig. 15 While the super-resolution texture estimate (**b**) already improves over the commonly used weighted average (**a**), the jointly estimated displacement map leads to a much more detailed result (**c**)



input image *input geometry* *initial error*

photometric error *geometric error*

Gradient descent after 4 and 8 iterations

Decoupling method after 4 and 8 iterations

Fig. 16 The graphs show the evolution of the photometric error with the number of iterations, for both the simple gradient descent scheme (*blue*) as well as our proposed decoupling framework (*red*). Photometric error is measured as the value of the energy (23), while geometric error is measured as the L^1-norm of the displacement field v. Although the photometric errors converge to similar values in both cases, the geometric error is much lower with the proposed decoupling method. This can be understood by looking at the evolution of the field v, where we can see that gradient descent gets stuck in a local minimum. In our proposed scheme the magnitude of the flow is overall much lower and converges steadily (Color figure online)

the texel-wise initialization from full-resolution input. This clearly demonstrates the viability of the super-resolution approach.

7.3 Texture and Displacement from Real-World Images

To demonstrate the merits of texture super-resolution reconstruction on real-world images, we performed experiments on three different datasets with 36 input images at input image resolution 768 × 584, see Fig. 18. Initial 3D reconstructions were obtained using an implementation of the algorithms in Kolev et al. (2007, 2010). After creating a conformal atlas, a texture map was computed using the proposed super-resolution algorithm, with the same texel-wise initialization as for the synthetic data sets.

The data is challenging for a texture reconstruction method, since the initial geometry and camera calibration is inaccurate, and weighted averaging of color information over

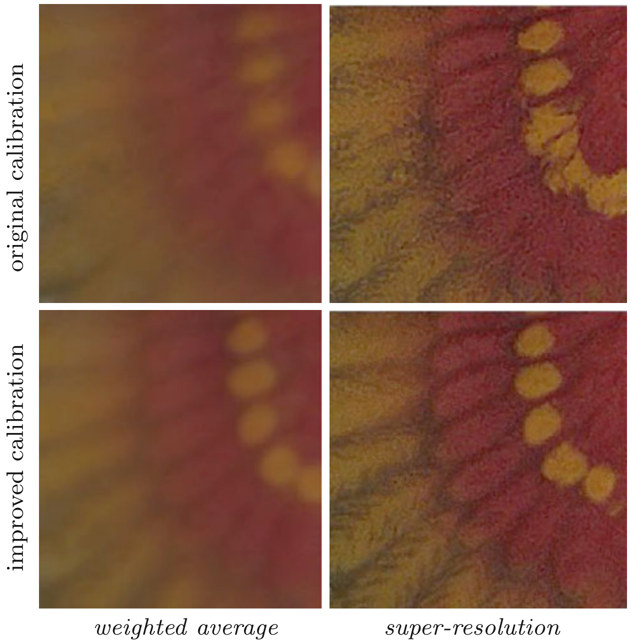


Fig. 17 Texture improvements after refining the camera parameters with the proposed approach for the bird data set, see Fig. 18. Displayed above are close-ups of a texture map obtained by averaging the input images (*left*) and a super-resolution texture map (*right*). Note the enhancement of the texture pattern as well as the removal of visual artifacts obtained with the refined calibration parameters

cameras leads to a blurry initialization, as can be observed in Fig. 15. Nevertheless, the optimized texture estimated via super-resolution already exhibits more details than an individual original input image, see Fig. 18. This is something no previous method for texture generation could hope for. The only visible artifacts result from large inaccuracies in 3D geometry, while no seams due to visibility boundaries can be observed.

When optimizing for small scale displacements using the proposed algorithm, the sharpness of the resulting texture increases considerably, see Fig. 15. A rendering of the final textured model is of at least similar quality than an input image from the same viewpoint, and usually the level-of-detail is even exceeded, see Fig. 18. The displacement and derived normal map can be leveraged to add effects into the rendering, like including additional light sources, as exemplified in Fig. 14. However, it should be noted that the displacement map is primarily a way to improve texture registration and thus sharpness and quality of the texture. In particular, it does not necessarily lead to an improved geometry in the sense that the displaced surface is closer to the ground truth shape of the object, since registration errors are often not only caused by deviations from the true geometry.

7.4 Improving Calibration

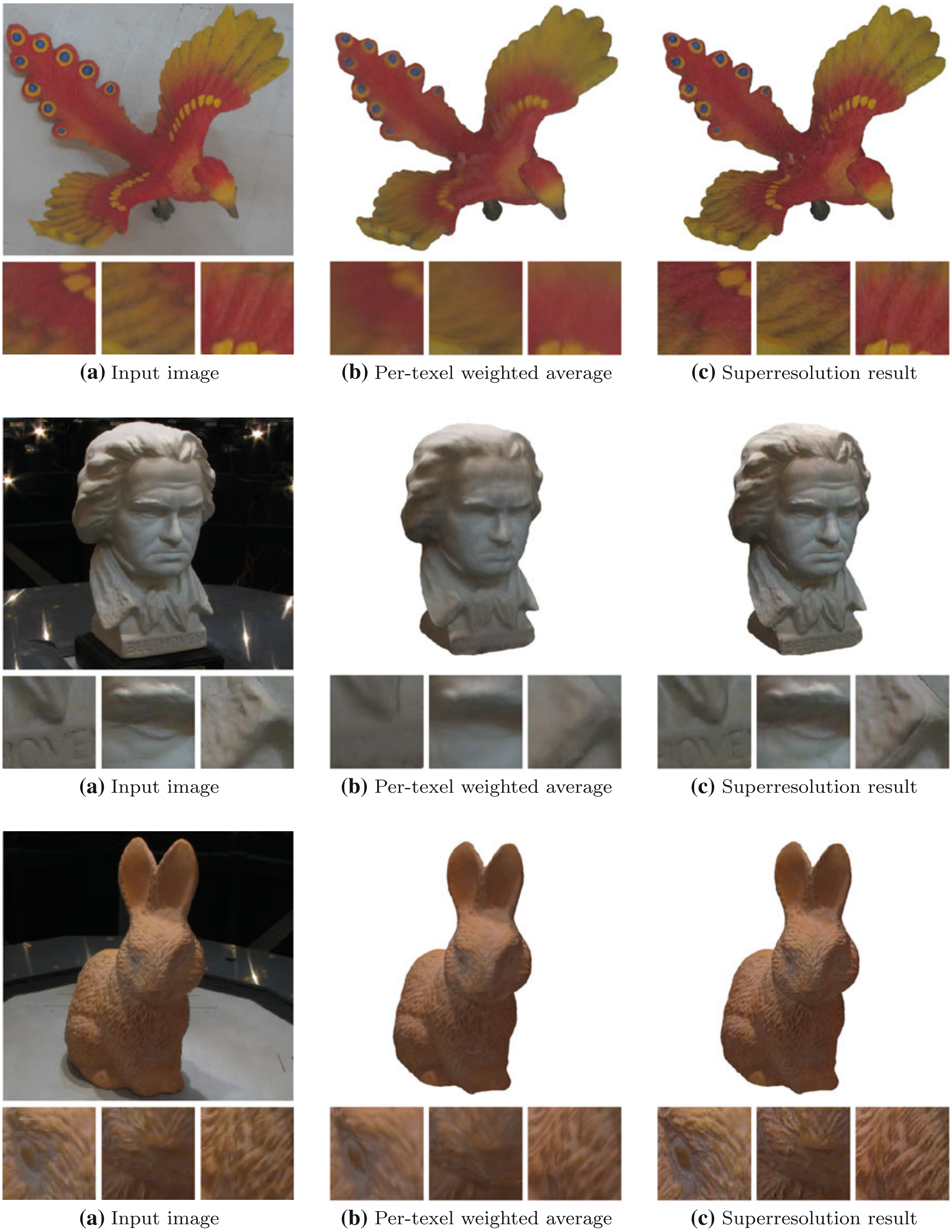
We evaluate the performance of the proposed calibration approach by examining the improvements in the photometric error and estimated texture map. In Fig. 16, we compare the results of optimization from the gradient descent algorithm and the proposed decoupling framework described in Sect. 6. We used an image sequence capturing a bunny figurine consisting of 36 views with manually distorted calibration parameters.[2] It can be observed that while the photometric error converges to a similar value for both methods, the geometric error, which is represented by the magnitude of the optical flow, converges to a substantially smaller value with the proposed decoupling scheme. Furthermore, the decoupled model leads to less outliers, a faster convergence rate and more resilience to local minima. Contrary to sparse feature-point based methods, the proposed formulation also has the advantage of not favoring any particular regions depending on the number of salient points. This is important to avoid scene-specific accumulation effects.

To show that our framework indeed improves the camera parameters, we compute a texture map for a fixed geometric model before and after applying the proposed calibration optimization, see Fig. 17. In order to be independent from the particular approach for texture estimation, we show comparisons for the naive averaging method as well as the super-resolution model. It can be observed that the simple color averaging technique produces quite blurry results. Nevertheless, the proposed calibration procedure alone already leads to a visible enhancement of the texture pattern. The improvements are even more notable when applying the super-resolution framework. In this case, visual artifacts due to scene geometry errors are removed, and we obtain a high-quality texture map. Note that the fine-scale texture details are clearly visible.

7.5 Limitations

As common in multi-view reconstruction, the current main limitation is that the surfaces need to be Lambertian. If the 3D model is accurate, then a good texture can still be computed if a surface point exhibits specularities in a few input images. However, the final color will essentially be averaged during optimization, so only information about the Lambertian component of the BRDF is recovered in the result. Obviously, a possible avenue for future work would be to extend our variational model to reconstruct not only diffuse color, but also information about reflectance properties from the data.

[2] The multi-view datasets in Fig. 18 are publicly available on our webpage, http://cvpr.in.tum.de.



(a) Input image **(b)** Per-texel weighted average **(c)** Superresolution result

(a) Input image **(b)** Per-texel weighted average **(c)** Superresolution result

(a) Input image **(b)** Per-texel weighted average **(c)** Superresolution result

Fig. 18 Results from three real-world multi-view datasets. The 3D model is rendered with the texture from texel-wise initialization (**b**) and the texture resulting from the proposed joint displacement map and super-resolution algorithm (**c**). The result has more visible details than an input image taken from the same viewpoint (**a**). The *rows below* the *large images* show some close-ups. The reader is invited to zoom in on the electronic version to better appraise the differences. See also Figs. 1 and 19 for larger close-ups

While the inverse problem for the texture is convex and thus requires no initialization, improving the local registration via geometry and calibration parameter optimization requires a reasonably close initialization of both. As a rule of thumb, both can only correct for initial reprojection errors in the range of a few pixels until the

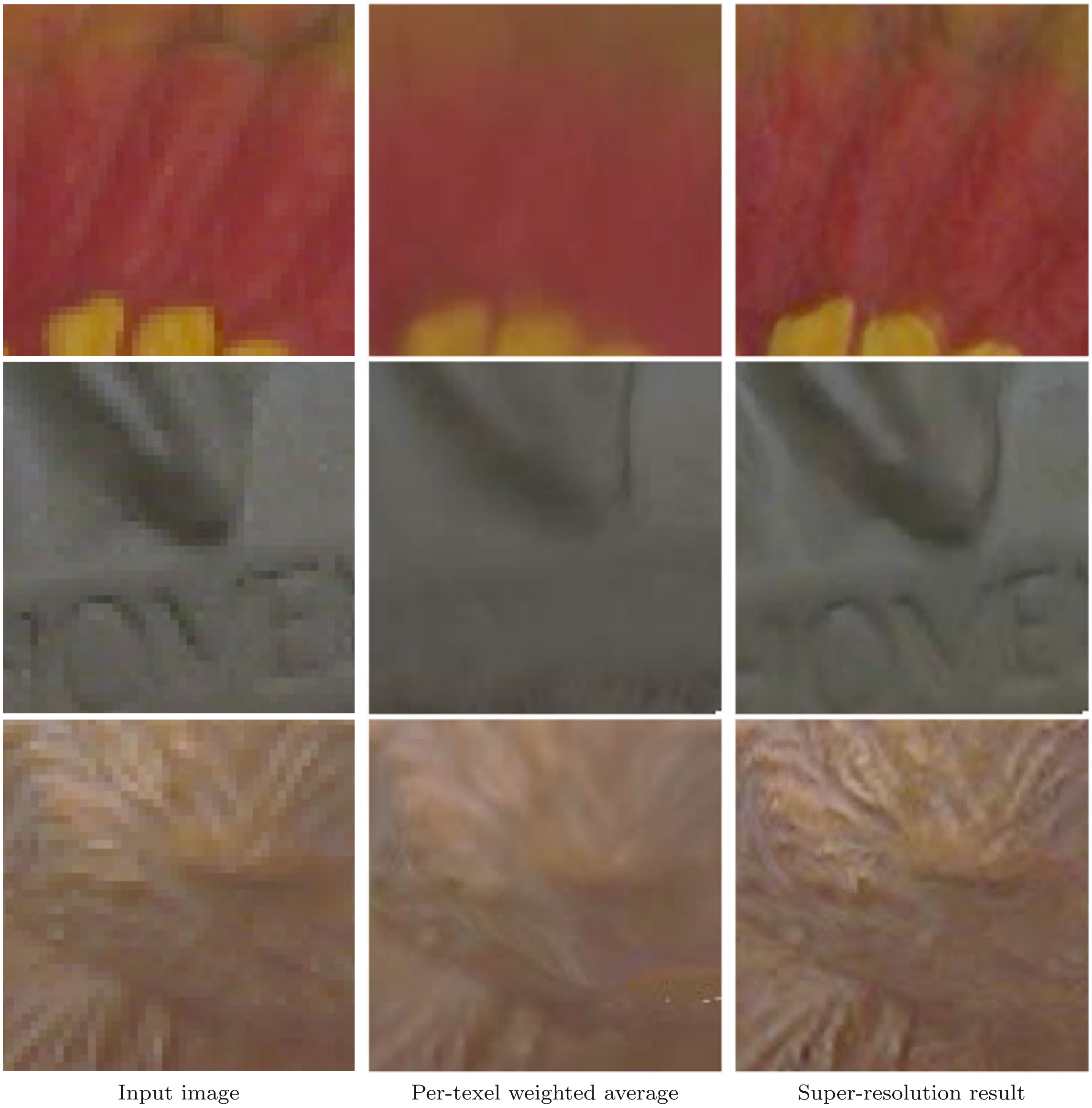


Fig. 19 Enlarged closeups for the results from three real-world multi-view datasets in Fig. 18. The improved details of the super-resolved texture compared to both input image as well as the texture obtained by averaging the input images are clearly visible

distortion becomes too large to recover from. However, state-of-the art multi-view stereo reconstruction today usually achieves results which are easily good enough for this purpose.

8 Conclusion

We proposed a super-resolution approach to multi-view reconstruction, and in particular, we believe that this is the first super-resolution model for curved surfaces. Starting from a single variational model for super-resolution image formation, we derive algorithms to iteratively optimize for a high-quality texture, a surface displacement map, and camera calibration parameters in order to improve the registration of the individual images on the surface.

The model is transformed to planar 2D texture space using a conformal atlas to facilitate optimization. The super-resolution texture estimation is a convex problem which we solve optimally using state-of-the-art convex optimization techniques. The displacement and calibration estimation problems are not convex. We therefore employ relaxation techniques allowing us to solve these problems by means of sequential convex programming.

The method produces high-fidelity texture maps exhibiting a level-of-detail and crispness well beyond that of individual input images. Experiments on several real-world objects demonstrate that both the resulting displacement map as well as the refined camera parameters provided by our method lead to a significant further improvement of the estimated super-resolved textures (Fig. 19).

Acknowledgments We thank Martin R. Oswald for providing the visualization in Fig. 1. This work was supported by the ERC Starting Grant "Convex Vision".

References

Allne, C., Pons, J. P., & Keriven, R. (2008). Seamless image-based texture atlases using multi-band blending. In *19th International Conference on Pattern Recognition*.

Attouch, H., Buttazzo, G., & Michaille, G. (2006). *Variational analysis in Sobolev and BV spaces. MPS-SIAM series on optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics.

Aubry, M., Kolev, K., Goldluecke, B., & Cremers, D. (2011). Decoupling photometry and geometry in dense variational camera calibration. In *Proceedings of ICCV*.

Baker, S., & Kanade, T. (2002). Limits on super-resolution and how to break them. *PAMI*, *24*(9), 1167–1183.

Bernardini, F., Martin, I., & Rushmeier, H. (2001). High-quality texture reconstruction from multiple scans. *IEEE Transactions on Visualization and Computer Graphics*, *7*(4), 318–332.

Bertalmio, M., Sapiro, G., Cheng, L. T., & Osher, S. (2001). Variational problems and PDEs on implicit surfaces. In *Proceedings of the IEEE Workshop on Variational and Level Set Methods* (*VLSM'01*) (pp. 186–193).

Bhat, P., Zitnick, C., Snavelny, N., Agarwala, A., Agrawala, M., Cohen, M., et al. (2007). Using photographs to enhance videos of a static scene. In *Eurographics Symposium on Rendering*.

Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J. P., & Osher, S. (2007). Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and Vision*, *28*, 151–167.

Capel, D., & Zisserman, A. (2001). Super-resolution from multiple views using learnt image models. In *Proceedings of CVPR* (Vol. 2, pp. 627–634).

Chambolle, A., & Pock, T. (2011). A first-order primal–dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, *40*(1), 120–145.

Donnelly, W. (2005). *GPU Gems 2, chapter per-pixel displacement mapping with distance functions*. Amsterdam: Addison-Wesley Longman.

Eisemann, M., Decker, B. D., Magnor, M., Bekaert, P., de Aguiar, E., Ahmed, N., et al. (2008). Floating textures. *Computer Graphics Forum (Proceedings of Eurographics)*, *27*(2), 409–418.

Floater, M. S., & Hormann, K. (2006). Surface parameterization: A tutorial and survey. In *Advances in multiresolution for geometric modelling*, *mathematics and visualization* (pp. 157–168). Berlin: Springer.

Fransens, R., Strecha, C., & van Gool, L. (2007). Optical flow based super-resolution: A probabilistic approach. *Computer Vision and Image Understanding*, *106*(1), 106–115.

Furukawa, Y., & Ponce, J. (2009). Accurate camera calibration from multi-view stereo and bundle adjustment. *International Journal of Computer Vision*, *84*, 257–268.

Goldluecke, B., & Cremers, D. (2009a). A superresolution framework for high-accuracy multiview reconstruction. In *Pattern Recognition* (*Proceedings of DAGM*).

Goldluecke, B., & Cremers, D. (2009b). Superresolution texture maps for multiview reconstruction. In *Proceedings of ICCV*.

Goldluecke, B., Strekalovskiy, E., & Cremers, D. (2012). The natural total variation which arises from geometric measure theory. *SIAM Journal on Imaging Sciences*.

Gu, X., & Yau, S.T. (2003). Global conformal surface parameterization. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (Vol. 43, pp. 127–137).

Hartley, R., & Zisserman, A. (2004). *Multiple view geometry in computer vision* (2nd ed.). Cambridge, MA: Cambridge University Press.

Jin, H., Cremers, D., Wang, D., Yezzi, A., Prados, E., & Soatto, S. (2008). 3-D reconstruction of shaded objects from multiple images under unknown illumination. *International Journal of Computer Vision*, *76*(3), 245–256.

Klein, G., & Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *6th IEEE and ACM International Symposium on Mixed and Augmented Reality* (*ISMAR*) (pp. 225–234). http://ewokrampage.wordpress.com/.

Kolev, K., Klodt, M., Brox, T., & Cremers, D. (2009). Continuous global optimization in multiview 3D reconstruction. *International Journal of Computer Vision*, *84*(1), 80–96.

Kolev, K., Klodt, M., Brox, T., Esedoglu, S., & Cremers, D. (2007). Continuous global optimization in multiview 3D reconstruction. In *Energy Minimization Methods in Computer Vision and Pattern Recognition* (*EMMCVPR*) (Vol. 4679, pp. 441–452).

Kolev, K., Pock, T., & Cremers, D. (2010). Anisotropic minimal surfaces integrating photoconsistency and normal information for multiview stereo. In *European Conference on Computer Vision* (*ECCV*), Heraklion, Greece.

Lempitsky, V., & Ivanov, D. (2007). Seamless mosaicing of image-based texture maps. In *Proceedings of CVPR* (Vol. 1, pp. 1–6).

Lensch, H., Heidrich, W., & Seidel, H. P. (2001). A silhouette-based algorithm for texture registration and stitching. *Graphical Models*, *63*(4), 245–262.

Lévy, B., Petitjean, S., Ray, N., & Maillot, J. (2003). Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics (SIGGRAPH)*, *21*(3), 362–371.

Lui, L. M., Wang, Y., & Chan, T. F. (2005) Solving PDEs on manifold using global conformal parameterization. In *Variational*, *Geometric, and Level Set Methods in Computer Vision*: *Third International Workshop* (*VLSM*) (pp. 309–319).

Mitzel, D., Pock, T., Schoenemann, T., & Cremers, D. (2009). Video super resolution using duality based TV-L1 optical flow. In *Pattern Recognition* (*Proceedings of DAGM*), Jena, Germany.

Pock, T., Cremers, D., Bischof, H., & Chambolle, A. (2009). An algorithm for minimizing the piecewise smooth Mumford–Shah functional. In *Proceedings of ICCV*, Kyoto, Japan.

Pock, T., Schoenemann, T., Graber, G., Bischof, H., & Cremers, D. (2008). A convex formulation of continuous multi-label problems. In *Proceedings of ECCV*.

Rav-Acha, A., Kohli, P., Rother, C., & Fitzgibbon, A. (2008). Unwrap mosaics: A new representation for video editing. *Proceedings of the ACM SIGGRAPH*, *27*(3), 17–25.

Rudin, L. I., Osher, S., & Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D*, *60*, 259 –268.

Schoenemann, T., & Cremers, D. (2008). High resolution motion layer decomposition using dual-space graph cuts. In *Proceedings of CVPR* (pp. 1–7).

Seitz, S., Curless, B., Diebel, J., Scharstein, D., & Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of CVPR* (pp. 519–528).

Snavely, N., Seitz, S., & Szeliski, R. (2006). Phototourism: Exploring image collections in 3D. In *Proceedings of the ACM SIGGRAPH*. http://phototour.cs.washington.edu/bundler/.

Sroubek, F., Cristobal, G., & Flusser, J. (2007). A unified approach to superresolution and multichannel blind deconvolution. *IEEE Transactions on Image Processing*, *16*(9), 2322–2332.

Stam, J. (2003). Flows on surfaces of arbitrary topology. *ACM Transactions on Graphics (SIGGRAPH)*, *22*(3), 724–731.

Strecha, C., von Hansen, W., Gool, L. V., Fua, P., & Thoennessen, U. (2008). On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proceedings of CVPR*.

Theobalt, C., Ahmed, N., Lensch, H., Magnor, M., & Seidel, H. P. (2007). Seeing people in different light-joint shape, motion, and reflectance capture. *IEEE Transactions on Visualization and Computer Graphics*, *13*(4), 663–674.

Triggs, B., McLauchlan, P., Hartley, R., & Fitzgibbon, A. (2000). Bundle adjustment—A modern synthesis. In *Vision algorithms*: *Theory and practice*, *lecture notes in computer science* (Vol. 1883, pp. 298–372). Berlin: Springer.

Unal, G., Yezzi, A., Soatto, S., & Slabaugh, G. (2007). A variational approach to problems in calibration of multiple cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*, 1322–1338.

Wang, Y., Gu, X., Hayashi, K., Chan, T. F., Thompson, P., & Yau, S. T. (2005). Surface parameterization using Riemann surface structure. In *Proceedings of ICCV* (Vol. 2, pp. 1061–1066).

Wanner, S., & Goldluecke, B. (2012). Spatial and angular variational super-resolution of 4D light fields. In *Proceedings of ECCV*.

Zach, C., Pock, T., & Bischof, H. (2007). A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition* (*Proceedings of DAGM*) (pp. 214–223).