The final version is available at https://doi.org/10.1016/j.cag.2013.12.001

High-Resolution Depth for Binocular Image-Based Modelling

David Blumenthal-Barby[†], Peter Eisert[††]

[†]*Fraunhofer Heinrich Hertz Institute, Berlin, Germany*
[††]*Humboldt Universität zu Berlin, Germany*

Abstract

We propose a binocular stereo method which is optimized for reconstructing surface detail and exploits the high image resolutions of current digital cameras. Our method occupies a middle ground between stereo algorithms focused at depth layering of cluttered scenes and multi-view "object reconstruction" approaches which require a higher view count. It is based on global non-linear optimization of continuous scene depth rather than discrete pixel disparities. We propose a mesh-based data-term for large images, and a smoothness term using robust error norms to allow detailed surface geometry. We show that the continuous optimization approach enables interesting extensions beyond the core algorithm: Firstly, with small changes to the data-term camera parameters instead of depth can be optimized in the same framework. Secondly, we argue that our approach is well suited for a semi-interactive reconstruction work-flow, for which we propose several tools.

Keywords: 3D reconstruction, image-based modelling, optimization, interaction in 3D reconstruction

1. Introduction

Binocular stereo algorithms compute a depth map from a pair of photographs or video frames. Driven by benchmarks used in the Computer Vision community [1] and applications like driver assistance, the leading algorithms are optimized for recovering depth-layers of cluttered scenes and precise object boundaries. Often, they use a finite set of disparities, and therefore the resulting depth maps show little surface detail. Operating on the pixel grid, many algorithms are limited to low image resolutions, especially if high quality global optimization methods are used. However, graphics applications like image-based modelling, photo relighting, or the fabrication of physical models with 3D printing require detailed surface geometry rather than depth layering of cluttered scenes. Surface meshes with impressive detail can be computed with state of the art multi-view algorithms. Unsurprisingly, however, those rely heavily on the availability of a large number of views.

In this paper, we propose an algorithm between these poles—a binocular stereo method optimized for computing detailed surface geometry. It exploits image resolutions in the 10 to 20 megapixel range which is typical for today's digital cameras. Combined with self calibration, our approach enables high quality "walk-along" stereo on pairs of casual images shot free-hand, a few footsteps apart.

Like many binocular stereo algorithms, we use global energy minimization in a "data-term / smoothness-term" framework. However, we deviate from typical binocular stereo schemes in several respects, using strategies more common in multi-view methods: We use a triangle mesh in the image plane to decouple the number of variables from the number of pixels, allowing us to exploit high image resolutions *and* global optimization. Similar to patch-based reconstruction and surface evolution methods, continuous depth parametrization is used instead of discrete disparities, which has several advantages: Most importantly, it does neither restrict surface detail nor impose a tradeoff between detail and computational complexity due to a larger label set. It also allows us to omit image rectification, which is beneficial for casual free-hand stereo where camera rotation between the shots induces strong distortions in the rectification. Finally, we use a versatile Gauss-Newton-type optimizer which is straightforward to implement on top of widely available numerical software libraries. These topics are addressed in the first part of the paper (section 3).

In the second part we outline two extensions of the core method, demonstrating the versatility of the proposed approach. We show that with small modifications to the energy function we can optimize camera parameters instead of depth in the same framework, which is useful to correct errors in calibration (sec. 4.1). Secondly, we describe a set of interactive tools which allow the user to influence or correct the reconstruction. User interaction in 3D reconstruction is a little-discussed topic, despite of the great practical success of intelligent semi-interactive tools for many tasks in graphics, such as segmentation, image retargeting or camera tracking. We argue that our approach is well suited for an interactive "optimize—adjust—re-optimize" work-flow, similar to semi-automatic segmentation (sec. 4.2).

Finally, we address initialization and convergence, and present results which we compare to other methods (sec. 5).

*Einsteinufer 37, 10827 Berlin, Germany

Email address: peter.eisert@hhi.fraunhofer.de
(David Blumenthal-Barby[†], Peter Eisert[††])

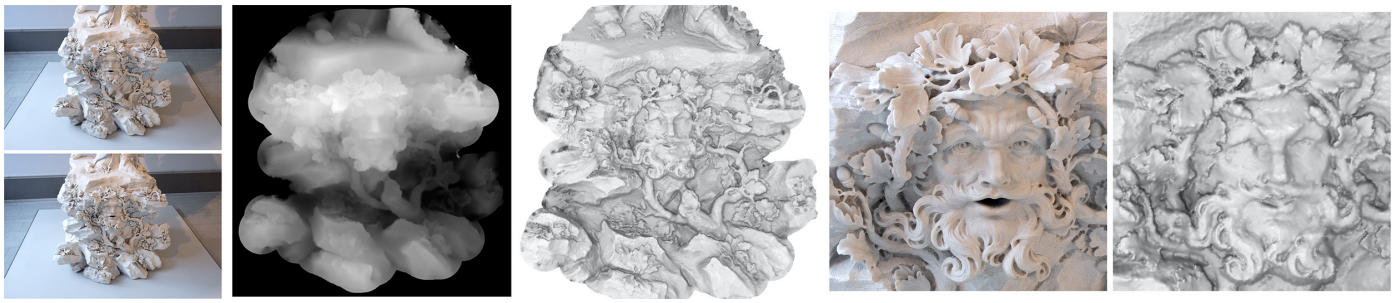


Figure 1: Left to right: (1) Complete set of input images, 12 megapixels, shot freehand in a museum under available light; (2) depth map representation of our reconstruction; (3) reconstruction rendered as shaded mesh; (4,5) detail crop of the left view and the shaded reconstruction.

2. Related work

2.1. Binocular stereo

A taxonomy and review of classic stereo algorithms is given by [1]. In the following we outline the differences between the focus of typical binocular stereo methods and the scope of our algorithm. Most research in binocular stereo is aimed at the reconstruction of scenes with multiple objects and a complex cluttered depth structure, prototypically represented by the Middlebury stereo benchmarks [1, 2]. This prevalent scene type influences the strategies employed by the algorithms: For example, color similarity is often used as an indicator for depth continuity [3, 4, 5]. Combining scene segmentation with stereo has proven to be highly effective, such that algorithms like [6] perform these tasks in a joint optimization.

For detailed reconstruction of objects, which is the focus of our work, these scene priors often do not apply. Color information or segmentation, for example, cannot be used for a detailled reconstruction of artifacts like the one show in figure 1.

Most stereo algorithms assume image pairs to be rectified such that correspondences can be searched along scan lines. This enables them to represent depth by disparity, measured in pixels, rather than by distance in a world coordinate system. As the pixel grid is discrete, many algorithms operate on a finite set of disparity values and use discrete global optimization methods such as Graphcuts and its descendants [6], belief propagation [4, 5] or other message passing methods [7] . While this yields impressive results for the scenes these algorithms aim for, it limits the amount of surface detail for object reconstruction. This can be relaxed to some degree by using subpixel disparities (e.g. [8]) at the cost of a larger label set, or by additional refinement steps after the reconstruction process (e.g. [9, 10]). Our method, like many multi-view approaches, is based on continuous depth parametrization and continuous optimization, which has the advantage of not limiting detail.

2.2. Multi-view reconstruction

Detailed surface reconstruction is the domain of multi-view methods which compute a mesh or volume representation of the scene rather than a depth map. As a comprehensive review of this area is beyond scope, we focus this section mostly on the relation of our work to multi-view approaches which use depth-maps computed from a small number of images from similar viewpoints at an intermediate stage, merging them later to obtain the final mesh. Examples of these methods are [10, 9, 7, 11].

The most significant conceptual difference is that multi-view methods, even if based on intermediate depth maps, rely on the availability of more than two views. The larger number of views can be exploited in several ways. Most importantly, it provides more image data to verify correspondence hypotheses based on photometric consistency. This makes the estimation of depth (and, sometimes, normals as in [12]) significantly more reliable than in the binocular case. Due to this increased matching robustness, many algorithms omit costly global smoothness terms which are used in binocular stereo and compute stereo matches locally [10, 12, 9, 11] . Smoothness is enforced later in outlier filtering, meshing or refinement stages. On the opposite side, surface evolution approaches like [13, 14] use a single continuous global optimization on a highly sophisticated error functional that models surface visibility over multiple views in a mathematically precise manner. These methods, however, are quite involved with respect to numerics.

Some methods (e.g. [9]) restrict depth computation to binocular view pairs. While these approaches do not exploit the larger view count in the matching stage of the algorithm, the redundant scene coverage provided by multiple views allows them to be very strict in filtering out potential mismatches without risking holes in the overall reconstruction. In contrast, optimization-based binocular algorithms such as ours employ smoothness terms to propagate information into areas of low matching confidence. Many multi-view methods, including, for example, [9, 12], also use the visual hull of the object as a constraint or to filter outliers. This is not possible in the binocular case.

We conclude this section by pointing out three specific related works. Firstly, an interesting but specialized stereo approach is used by [10] for face reconstruction. They describe a complex iterative process of strictly local matching, filtering, and refinement steps on an image pyramid which produces detailed depth maps of the face in the binocular stereo stage. The impressive results of their overall system are based on multi-view data and on a face-specific approach to detail enhancement.

Secondly, there is an interesting connection of our work to Patch-Based Multiview Stereo (PMVS) [12], which is still one

of the most successful multi-view approaches. PMVS uses a three stage approach. First, corner points are matched across images to obtain an initial sparse point cloud. Second, depth together with surface normals is computed using nonlinear optimization on isolated patches, using the detected points as seed locations. Then, a closed surface is computed, which is refined by repeatedly optimizing patches located at the surface's vertices, smoothness, and consistency with the scenes silhouettes. Our method shares the initialization stage, where we use SIFT [15] instead of Harris corners. Then we directly optimize the reprojection of a connected triangle mesh, treating the mesh triangles as patches, where the connectivity induced by the mesh enables us to use small triangles. We also employ smoothness terms, which are critical to our approach due to the low number of views.

Finally, [16] proposed an interactive multi-view reconstruction system where the user guides the entire reconstruction process manually by growing a mesh over time with a paint-like tool. The reconstruction process, which runs on the GPU while the user paints, is based on an optimization similar to ours. There are, however, significant differences between their approach and ours in both intent and algorithm. Most importantly, the focus of our paper is on getting the most detail out of binocular high-resolution input, for which we propose a linearized data-term to cope with the large number of pixels, a mixed first and second order smoothness term for detailed geometry, and a method for photometric adaptation. The system of [16] is focused on multi-view input. Therefore it has to handle visibility issues, but uses simpler data and smoothness terms. Due to the user-guided reconstruction process, their optimization can be restricted to the vicinity of the user's brush tool, whereas our method optimizes globally over the entire high resolution image.

We fully agree with [16] that putting the user in the loop is important for practical 3D reconstruction. In section 4.2 we address several tools for an interactive reconstruction work-flow. While in [16] the user guides the optimization process itself, our tools focus on an "optimize—adjust—re-optimize" workflow similar to semi-automatic segmentation.

3. Continuous stereo

3.1. Optimization framework

Our stereo algorithm is based on a nonlinear image-based optimization approach which was originally developed in the context of tracking [17]. The geometric relation between two single-channel images $\mathcal{I}$ and $\mathcal{J}$ is described by a warp function $\mathcal{W} : \mathbb{R}^2 \times \mathbb{R}^K \to \mathbb{R}^2$. Conceptually, the assumed image relation for all pixel coordinates $\mathbf{x}$ is:

$$\forall \mathbf{x} : \qquad \mathcal{I}(\mathbf{x}) = \mathcal{J}(\mathcal{W}(\mathbf{x}, \mathbf{d})) \tag{1}$$

The warp is determined by an unknown parameter vector $\mathbf{d} \in \mathbb{R}^K$, which is the quantity to be estimated. Note that the above equation presupposes that the images can be mapped onto each other by geometric deformation alone, i.e. without a change in pixel values. Relaxing this brightness constancy assumption is important for stereo and will be addressed in section 3.4.

In the following we briefly recapitulate how $\mathbf{d}$ can be estimated from an image pair. This section is independent of a specific warp function, which we describe in sect. 3.2 for stereo and sect. 4.1 for calibration. We first define the warp residual for a pixel:

$$r(\mathbf{x}, \mathbf{d}) = \mathcal{I}(\mathbf{x}) - \mathcal{J}(\mathcal{W}(\mathbf{x}, \mathbf{d}))$$

Estimating $\mathbf{d}$ amounts to minimizing the energy

$$\mathcal{E}(\mathbf{d}) = \sum_{\mathbf{x}} \rho(r(\mathbf{x}, \mathbf{d}))$$

where ρ is a norm-like function. This is the data-term of the optimization. We chose the Huber function [18] for ρ which switches from quadratic (L2) to linear (L1) influence above a threshold computed dynamically from the Median Absolute Deviation of the residuals. This reduces the relative influence of low confidence pixel matches, leading to a stronger local influence of the smoothness terms. For the stereo problem, this helps to prevent erratic geometry at occlusions and specular highlights.

For minimizing energies of this kind with arbitrary norm-like cost functions a Gauss-Newton type algorithm was developed in the statistics community by Wedderburn and McCullagh [19, 20], which is summarized in the appendix. In order to assess the computational costs of specific warp functions we briefly summarize its application to warp estimation. The Gauss-Newton update $\Delta\mathbf{d}$ is found by solving:

$$\left(\mathbf{J}_{\mathcal{E}}^T \mathbf{W} \mathbf{J}_{\mathcal{E}}\right) \Delta\mathbf{d} = -\mathbf{J}_{\mathcal{E}}^T \mathbf{W} \mathbf{r} \tag{2}$$

$\mathbf{W}$ is a diagonal matrix whose elements $\psi(\mathbf{x}, \mathbf{d})$ depend on ρ:

$$\psi(\mathbf{x}, \mathbf{d}) = \frac{1}{2} r(\mathbf{x}, \mathbf{d})^{-1} \frac{\partial \rho(r(\mathbf{x}, \mathbf{d}))}{\partial r}$$

ψ is called the weight-function of ρ as it scales the influence of individual equations in the normal equations. For details we refer to the references cited above. $\mathbf{J}_{\mathcal{E}}$ is the Jacobian[1] of $\mathcal{E}$, and its rows are the gradients ∇r^T of the residuals:

$$\nabla r^T = -\left(\nabla \mathcal{J}|_{\mathcal{W}(\mathbf{x},\mathbf{d})}\right)^T \cdot \mathbf{J}_{\mathcal{W}}|_{\mathbf{x},\mathbf{d}}$$

$\nabla \mathcal{J}|_{\mathcal{W}(\mathbf{x},\mathbf{d})}$ is the gradient of the second image, evaluated at the warped location. $\mathbf{J}_{\mathcal{W}}$ is the Jacobian of the warp function; its size is $2 \times K$ where K is the number of warp parameters. The size of $\mathbf{J}_{\mathcal{E}}$ is $N \times K$ where N is the number of pixels, and the linear system to solve is of size $K \times K$. Hence computing the elements of $\mathbf{J}_{\mathcal{E}}$ and $\mathbf{W}$ depends on the pixel count *and* the number of warp parameters, while the time to solve the normal equations depends only on the number of parameters. Solving eq. (2) is the dominant computational load for warp functions with high parameter count.

[1] Strictly speaking, $\mathbf{J}_{\mathcal{E}}$ is only the Jacobian of $\mathcal{E}$ for $\rho(r) = r^2$. The elegance of the iterative re-weighting algorithm of [19] lies in the fact that $\mathbf{J}_{\mathcal{E}}$ can be computed as if this were the case even if a different norm-like function is used.

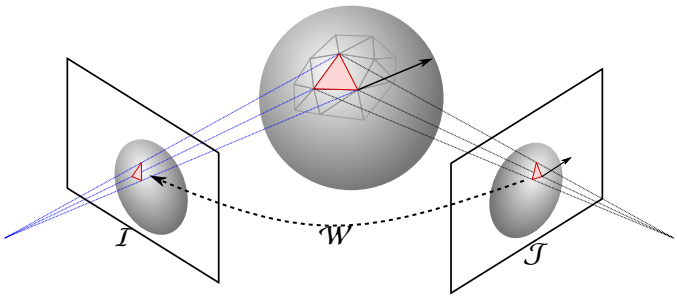


Figure 2: The warp function $\mathcal{W}$ describes the mapping of the second stereo image $\mathcal{J}$ onto the first one, $\mathcal{I}$. It is parametrized by the depths of the mesh vertices along their projection rays (blue lines).

We conclude this section by pointing out the computational efficiency of linear warp functions, which can be expressed canonically as matrix-vector product $\mathcal{W}(\mathbf{x}, \mathbf{d}) = \mathbf{M}_\mathbf{x}\mathbf{d}$. The subscript $\mathbf{x}$ indicates that $\mathbf{M}_\mathbf{x}$ can vary for every pixel coordinate $\mathbf{x}$. For this warp, the Jacobian $\mathbf{J}_\mathcal{W}|_{\mathbf{x},\mathbf{d}} = \mathbf{M}_\mathbf{x}$ is constant with respect to $\mathbf{d}$ and can be precomputed. In this case, only the image gradient $\nabla\mathcal{J}$ and the weight matrix $\mathbf{W}$ varies over the Gauss-Newton iterations.

3.2. A stereo warp function

In order to perform global optimization on images in the 10 to 20 megapixel range, we formulate our stereo warp function on a triangle mesh rather than on the pixel grid. This allows us to chose the geometric resolution, which determines K, to be lower than the image resolution and still exploit all pixels in the optimization. The mesh is generated in the first stereo image $\mathcal{I}$ using the algorithm of [21] and remains fixed throughout the optimization. This has the advantage that quantities depending on the mesh geometry need not be recomputed in every iteration. Assuming that the scene is piecewise flat between the mesh vertices, the remaining degree of freedom is the depth of each 3D vertex along a projection ray through the camera center and the corresponding fixed 2D vertex, as illustrated in figure 2. These depths are the the warp parameters $[d_1 \dots d_K]^T = \mathbf{d}$ we optimize for.

The relation between each mesh triangle in $\mathcal{I}$ and its corresponding triangle in $\mathcal{J}$ is a homography. However, computing the warp Jacobian of a homography for every pixel in every iteration is expensive due to its non-linearity. In order to exploit the advantages of linear warp functions described above we linearize the pixel motion, i.e. we approximate the homographies by affinities. The affinities are induced by the mesh vertices which move in 3D space along the projection rays, as described above. In the following we describe this warp function formally.

Let (u_i, v_i) denote the 2D location of the ith fixed mesh vertex in image $\mathcal{I}$. We concatenate the coordinates of all vertices in two vectors $\mathbf{u} = \begin{bmatrix} u_1 \dots u_K \end{bmatrix}^T$ and $\mathbf{v} = \begin{bmatrix} v_1 \dots v_K \end{bmatrix}^T$. For each pixel location $\mathbf{x} \in \mathbb{R}^2$ we define a sparse vector $\mathbf{b}_\mathbf{x} \in \mathbb{R}^K$ which contains the three barycentric coordinates of $\mathbf{x}$ with respect to the vertices of its surrounding triangle. We can now express $\mathbf{x}$ in terms of $\mathbf{u}$ and $\mathbf{v}$ as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{b}_\mathbf{x}^T & \\ & \mathbf{b}_\mathbf{x}^T \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \tag{3}$$

Note that $\mathbf{b}_\mathbf{x}$ can be precomputed for all pixels since the mesh is fixed in $\mathcal{I}$. Next, we replace vectors $\mathbf{u}$ and $\mathbf{v}$ with a vertex motion model that reflects the geometry of the stereo problem. In the following let $\mathbf{K}_\mathcal{I} \begin{bmatrix} \mathbf{R}_\mathcal{I} \, \mathbf{t}_\mathcal{I} \end{bmatrix}$ and $\mathbf{K}_\mathcal{J} \begin{bmatrix} \mathbf{R}_\mathcal{J} \, \mathbf{t}_\mathcal{J} \end{bmatrix}$ denote the decompositions of the cameras into calibration, rotation and translation. The subscripts indicate the associated image, $\mathcal{I}$ or $\mathcal{J}$. For notational compactness we introduce a function $\mathcal{H} : \mathbb{R}^3 \to \mathbb{R}^2$ to describe the "de-homogenization" of a vector:

$$\mathcal{H}\left([x\ y\ w]^T\right) = \begin{bmatrix} \frac{x}{w} & \frac{y}{w} \end{bmatrix}^T$$

It is easily verified that

$$\mathcal{P}(d_i) = \begin{bmatrix} d_i \mathbf{R}_\mathcal{I}^{-1} \mathbf{K}_\mathcal{I}^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} - \mathbf{R}_\mathcal{I}^{-1} \mathbf{t}_\mathcal{I} \\ 1 \end{bmatrix} \tag{4}$$

describes a 3D point at depth d_i on the ray that projects to the image point $[u_i\ v_i]^T$ in $\mathcal{I}$, in homogeneous coordinates. The point in the second image $\mathcal{J}$ corresponding with $[u_i\ v_i]^T$ can now be described as a function of depth d_i along this ray as follows:

$$\begin{bmatrix} \mathcal{U}_i(d_i) \\ \mathcal{V}_i(d_i) \end{bmatrix} = \mathcal{H}\left(\mathbf{K}_\mathcal{J} \begin{bmatrix} \mathbf{R}_\mathcal{J} \, \mathbf{t}_\mathcal{J} \end{bmatrix} \mathcal{P}(d_i)\right) \tag{5}$$

Analogous to above, we concatenate the functions $\mathcal{U}_i$ and $\mathcal{V}_i$ into vectors $\boldsymbol{\mathcal{U}} = [\mathcal{U}_1 \dots \mathcal{U}_K]^T$ and $\boldsymbol{\mathcal{V}} = [\mathcal{V}_1 \dots \mathcal{V}_K]^T$. Substituting this model into equation (3) yields the stereo warp function:

$$\mathcal{W}(\mathbf{x}, \mathbf{d}) = \begin{bmatrix} \mathbf{b}_\mathbf{x}^T & \\ & \mathbf{b}_\mathbf{x}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\mathcal{U}}(\mathbf{d}) \\ \boldsymbol{\mathcal{V}}(\mathbf{d}) \end{bmatrix} \tag{6}$$

The warp Jacobian at pixel $\mathbf{x}$ and depth parameter $\mathbf{d}$ is:

$$\mathbf{J}_\mathcal{W} = \begin{bmatrix} \mathbf{b}_\mathbf{x}^T & \\ & \mathbf{b}_\mathbf{x}^T \end{bmatrix} \begin{bmatrix} \mathbf{J}_\mathcal{U}|_\mathbf{d} \\ \mathbf{J}_\mathcal{V}|_\mathbf{d} \end{bmatrix}$$

$\mathbf{J}_\mathcal{U}$ and $\mathbf{J}_\mathcal{V}$ are the Jacobians of the vertex motion model described by $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{V}}$. Each is of size $K \times K$ and diagonal, since the derivatives $\frac{\partial \mathcal{U}_i}{\partial d_j}$ and $\frac{\partial \mathcal{V}_i}{\partial d_j}$ are zero unless $i = j$. These Jacobians have to be re-computed in each iteration due to the non-linearity of $\mathcal{H}$. However, their size is proportional to the number of vertices and not to the number of pixels—this is the computational advantage of the linearized warp. We omit the statement of the analytical derivatives and note that bi-secant numerical derivatives work well in practice.

3.3. Smoothness terms

We use regularization terms in the energy function to control the smoothness of the depths. We have experimented with first and second order smoothness terms, with different norm-like functions applied to each, and with combinations of both types.

Our first order terms operate along the mesh edges. Let $\mathcal{N}(i)$ denote the set of neighbors of a vertex with index i, and

Figure 3: Effects of regularization terms: (1) first order L2; (2) first order Huber; (3) second order L2; (4) second order L2, weight doubled; (5) mixed first order L2 and second order Huber. Renderings are intentionally flat-shaded to emphasize the differences.

d_i the depth at this vertex, as above. Further let $e_{i,j}$ denote the length of the edge between vertices i and $j \in \mathfrak{N}_i$ in image $\mathcal{I}$. The first order smoothness terms are given by:

$$\mathcal{E}_{s1} = \sum_{i=1}^{K} \sum_{j \in \mathcal{N}(i)} \rho\left(e_{i,j}^{-1}\left(d_i - d_j\right)\right)$$

By setting $\rho(x) = |x|$ we can approximate a Total Variation L1 (TVL1) regularization. Using the Huber function gives a mixture between L1 and L2 regularization of depth differences.

Our second order terms are based on the cotangent Laplacian $\mathbf{L}$ of the reconstruction mesh. We compute $\mathbf{L}$ only once on the 2D mesh in $\mathcal{I}$, exploiting the fact that the mesh projection into $\mathcal{I}$ does not change. The second order term is given by:

$$\mathcal{E}_{s2} = \sum_{i=1}^{K} \rho(l_i) \quad \text{with} \quad [l_1 \dots l_K]^T = \mathbf{Ld} \tag{7}$$

Note that both smoothness terms have a constant Jacobian which can be precomputed: For $\mathcal{E}_{s1}$ the Jacobian is a vertex-vertex incidence matrix with two non-zero elements per row, for $\mathcal{E}_{s2}$ the Jacobian is $\mathbf{L}$. Therefore, only the weight matrix $\mathbf{W}$ has to be updated in the optimization.

Examples of the effects of different smoothness terms are shown in figure 3. First order regularization produces good but somewhat noisy results. Using the Huber function significantly reduces blurring of discontinuities. Laplace regularization is typically used with $\rho(x) = x^2$. The combination of Laplace and Huber leads to spike-shaped artifacts at discontinuities where the influence of the regularization is suppressed too strongly. This effect can be suppressed, however, with a weighted combination $\mathcal{E}_{s12} = \lambda\mathcal{E}_{s1} + (1-\lambda)\mathcal{E}_{s2}$ of first order regularization with L2 norm and Laplace regularization with Huber "norm", which produces smooth surfaces *and* good discontinuities. This is the smoothness term used for all results in the paper (with $\lambda = 0.5$).

3.4. Photometric adaptation

It is well known that the brightness constancy assumption does not hold for stereo problems. Therefore, most algorithms use similarity measures which are less sensitive to photometric variations than pixel difference. Examples are the widely used Normalized Cross Correlation (e.g. [9, 7, 11] and many more), Mutual Information [22] rank [2] or census [23]. These measures are often defined on image patches rather than individual pixels and they involve more or less complex, sometimes non-linear computations (e.g. rank, census). Some of them can be used in continuous optimization frameworks (e.g. [24]) but it complicates the derivatives and is computationally expensive. An alternative approach is based on the observation that high image frequencies are less affected by view-dependent lighting effects than low frequencies. Therefore, some form of high-pass filtering is applied to the images before reconstruction. While pre-filtering is computationally cheap, it discards potentially valuable image information. For our algorithm, we found this approach to have a negative effect on convergence unless filter parameters are fine-tuned individually for each image pair. A third approach is to include the estimation of lighting conditions or a model of luminance differences in the optimization. [25] use this strategy for multi-view reconstruction, [26] use it for tracking.

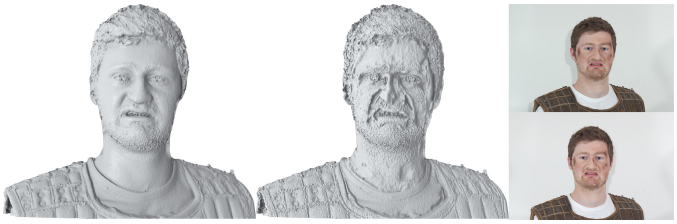


Figure 4: Reconstruction with (left) and without (center) photometric adaptation.

Our own approach is a computationally efficient combination of estimation and filtering. Conceptually, we replace the right-hand side of the brightness constancy assumption with a luminance residual term $\mathcal{L}^*(\mathbf{x})$:

$$\mathcal{I}(\mathbf{x}) - \mathcal{J}(\mathcal{W}(\mathbf{x}, \mathbf{d}^*)) = \mathcal{L}^*(\mathbf{x}) \tag{8}$$

$\mathcal{L}^*(\mathbf{x})$ describes the intensity difference between $\mathcal{I}$ and $\mathcal{J}$ at $\mathbf{x}$ under the optimal warp parameters $\mathbf{d}^*$. If $\mathbf{d}^*$ and hence $\mathcal{L}^*$ was known it could be subtracted from $\mathcal{I}$ and warp estimation could be computed on $\mathcal{I} - \mathcal{L}^*$ under the brightness constancy assumption. As $\mathcal{L}^*$ is unknown before the optimization terminates, we iteratively compute an approximation $\mathcal{L} \approx \mathcal{L}^*$ along with the geometric warp parameters $\mathbf{d}$. In contrast to the "photometric warp" of [26], we decouple warp estimation from the estimation of $\mathcal{L}$ in order to keep the size of the optimization problem constant. Let $\mathbf{d}^{(k)}$ be the state of the warp parameter in the kth Gauss-Newton iteration. We assume that the luminance

residual can be obtained from the low frequency content of the image residual $\mathcal{I}(\mathbf{x}) - \mathcal{J}(\mathcal{W}(\mathbf{x}, \mathbf{d}))$. This is the flip side of the high-frequency assumption the pre-filtering strategies are based on, but in contrast to pre-filtering, we adapt $\mathcal{L}$ iteratively to the current state of the optimization. In order to allow for discontinuities in $\mathcal{L}$ we use the guided filter [27], an edge-preserving smoothing operator similar to the bilateral filter. Once we have computed $\mathcal{L}$ we continue warp estimation with the following photometrically adapted data term:

$$\mathcal{E}(\mathbf{d}) = \sum_i \rho\left(\mathcal{I}(\mathbf{x}) - \mathcal{L}(\mathbf{x}) - \mathcal{J}(\mathcal{W}(\mathbf{d}, \mathbf{x}))\right)$$

The initial estimate of $\mathcal{L}$ is based on the initial warp parameters (see section 5) and computed before Gauss-Newton iteration starts. In practice we find it sufficient to re-estimate $\mathcal{L}$ every three to five iterations. Figure 4 illustrates the effect (and necessity) of luminance adaptation.

3.5. Parameters

The proposed method has a number of parameters that can—and have to—be adjust to obtain high quality results. In the following we briefly describe the most important ones.

- The *mesh resolution* determines the resolution of details visible in the result, the number of pixels that contribute to one vertex in the data term, and the size of linear system to be solved in the normal equation. Meshing is addressed in section 5.2.
- *Smoothness term weight.* The relative influence of the smoothness term against the data term is the most important parameter with respect to the look of the result. It is typically adjusted individually for each scene, depending on scene detail and image noise.
- *Norm-like function thresholds.* The thresholds in the Huber norm used in the data term as well as in the first-order smoothness terms are determined automatically using the Median Absolute Deviation (MAD) of the respective residuals. This methods has proven to be robust over a large variety of scenes.
- For *photometric adaptation* the kernel size of the guided filter used to separate the luminance residual from the warp residual must be adjusted to the image size.

4. Extensions

4.1. Camera warp

The use of depth parametrization and a generic continuous optimization framework makes it possible to estimate camera parameters instead of scene depth with very little change to the warp function. We only have to switch the parametrization such that depths d_i are treated as constants and camera parameters become variables. We can only estimate the relative pose of the second camera with respect to the first one. Therefore we keep $\mathbf{R}_\mathcal{I}$ and $\mathbf{t}_\mathcal{I}$ constant and parametrize eq. (4) with the intrinsic parameters of the first camera, if we want to include them in the calibration update at all. In (5), rotation $\mathbf{R}_\mathcal{J}$ and translation $\mathbf{t}_\mathcal{J}$ of the second camera become variables, as well as the intrinsics of the second camera if required. There are many ways of parametrizing rotation updates, e.g. by cross-product based linearization of Euler angles $\boldsymbol{\phi}$:

$$\mathbf{R}\mathbf{x} = \mathrm{rot}(\boldsymbol{\phi} + \Delta\boldsymbol{\phi})\,\mathbf{x} \approx \mathrm{rot}(\boldsymbol{\phi})\,\mathbf{x} + [\mathbf{x}]_\times \boldsymbol{\phi} \tag{9}$$

Denoting vectors of intrinsic camera parameters as $\mathbf{k}_\mathcal{I}$ and $\mathbf{k}_\mathcal{J}$, the vertex motion model (eq. (5)) is now $\mathcal{U}\left(\mathbf{k}_\mathcal{I}, \boldsymbol{\phi}_\mathcal{J}, \mathbf{t}_\mathcal{J}, \mathbf{k}_\mathcal{J}\right)$ and $\mathcal{V}\left(\mathbf{k}_\mathcal{I}, \boldsymbol{\phi}_\mathcal{J}, \mathbf{t}_\mathcal{J}, \mathbf{k}_\mathcal{J}\right)$. Note that the 3D vertices still project to the original 2D vertices of the mesh in $\mathcal{I}$, as before. Therefore the warp function (eq. (6)) and the optimization machinery remain unchanged except for the parameters. The size of $\mathbf{J}_\mathcal{U}$ and $\mathbf{J}_\mathcal{V}$ is now the number of vertices times the number of camera parameters to be estimated, and they are dense.

Our camera warp does not (yet) replace calibration in our reconstruction. It is sufficient, however, to correct the parameters of a pre-calibrated camera which was accidentally moved or refocused after calibration: We first run the stereo optimization. When the resulting depths are not satisfactory we optimize for depth and camera parameters in turns of five iterations. In principle, we can also optimize for depth and camera parameters simultaneously, which amounts to an image-based bundle adjustment. However, the sparse direct solver we currently use is slow on the arrow-shaped linear systems which occur in bundle adjustment problems. This can be improved with a specialized solver, e.g. [28].

4.2. Interactive tools

Most reconstruction algorithms, both multi-view and stereo, are black-box systems. In practical applications this forces users to manipulate appearance and correct geometry in generic 3D modeling packages. On the other hand, many state of the art graphics algorithms in segmentation, tracking or re-targeting enable user interaction, typically in the form of initialization and correction. We think that 3D reconstruction can profit from this kind of interactive work-flow, and we argue that the proposed algorithm is well suited for interactive tools. In this section we outline three examples.

4.2.1. Depth correction

Sometimes there are regions in an image pair where the optimization converges slowly or terminates prematurely due a local optimum. We provide a tool for adjusting depths in order to correct or speed up the reconstruction. It is important to note that the user's correction does not have to be exact as the optimization converges quickly to accurate depths when re-run with the user's approximate changes as starting point. This enables an "optimize, adjust, re-optimize" work-flow. Figure 5 illustrates an example of this process. We want to enable the user to correct depth in image space rather than in a 3D workspace which is non-intuitive for the untrained. We have experimented with several types of tools, such as adjusting individual vertex

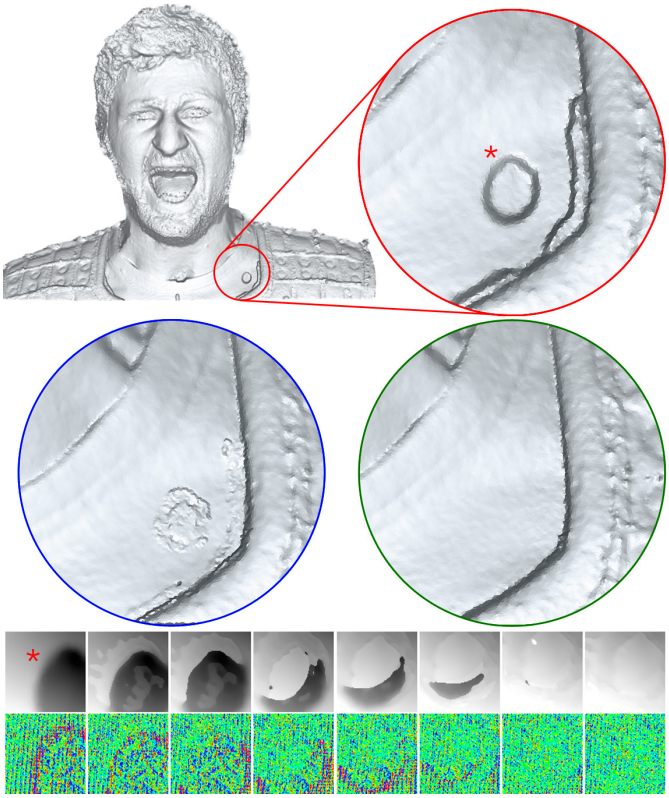


Figure 5: This reconstruction was halted after five iterations. Some regions have not converged yet, e.g. the one in the magnified detail highlighted with a red border. They are roughly corrected by the user with the proposed intelligent repair tool, as shown in the second magnification (blue border). Starting at the corrected depths, the optimization converges to a good result within two additional iterations (third magnification, green border). The bottom rows show the evolution of depth (first row) and pixel difference (second row, green is zero) while the user paints at the location marked with the red star.

correspondences, "sculpting" the reconstruction mesh, or painting on the depth map. We have learned that designing an efficient depth-correction is challenging, even when the correction does not have to be precise. Therefore we support correction computationally with a local stereo algorithm. This approach is inspired by [16], where, in contrast to this work, the entire global optimization is user guided.

Depth correction support works as follows: For each pixel under the paint tool, we store the best seen depth d^* and the last tested depth d; d^* is initialized with the result of the global optimization, and $d = d^*$ initially. When the user paints, we update $d \leftarrow d + \Delta d$ with Δd depending on the selected paint tool (e.g. add or subtract). After each update to d we compute the normalized cross correlation $NCC(d)$ of a patch around the pixel with the corresponding patch in the second image. We update $d^* \leftarrow d$ only if $NCC(d) > NCC(d^*)$. Finally, we smooth accepted depths d^* under the paint cursor with a 5×5 median filter. This process happens in real-time while the user paints. The evolution of depth during the paint process is illustrated in figure 5. After depth correction the vertex depths of the global optimization mesh are updated by sampling the d^*-depth map and the global optimization is re-run.

4.2.2. *Smoothness brush*

The weight of the second order (Laplacian) regularization term influences the smoothness of the reconstruction. By adjusting this term locally we can provide a tool which allows the user to express preference for a smoother or rougher result in parts of the image. Since the meshes are generated in image space the user can paint with a smoothness brush directly on the image. Before reconstruction, we sample this smoothness map at the mesh vertices and re-weight the corresponding rows of the Laplacian in the second order smoothness term accordingly. Again we can provide an iterative work-flow. When a region reconstructed as "rough" is to be made smoother we pre-smooth it strongly before re-running the optimization. The reason is that rough reconstructions have a lower residual as they give more influence to the data term which can prevent the optimization from converging to a smoother state.

4.2.3. *Interactive meshing*

Mesh construction is another aspect for which we provide interactive tools. A region of interest (ROI) tool allows the user to define the region of the image where the mesh is to be constructed. Additionally, she can specify paths in the image where she wants edges to be in the mesh. This is useful at strong discontinuities when coarse reconstruction meshes are used. For fine meshes, predefining edges is not necessary.

5. System overview and results

In this section, the overall warp-based stereo system is described. The following sections describe initialization and mesh generation. An overview of the data flow and the algorithms involved is shown in figure 7. Finally, results and comparison with other methods are discussed.

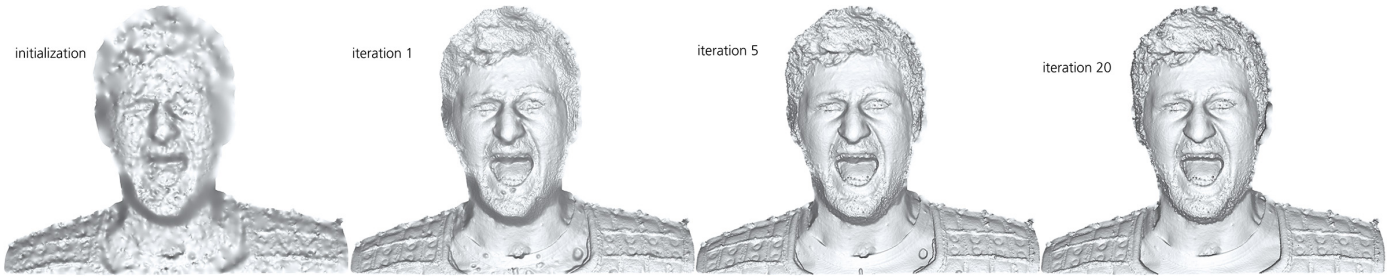


Figure 6: Initialization from sparse interest point correspondences (top left) and convergence over 20 iterations. Many surface details appear in the first iteration (top right).

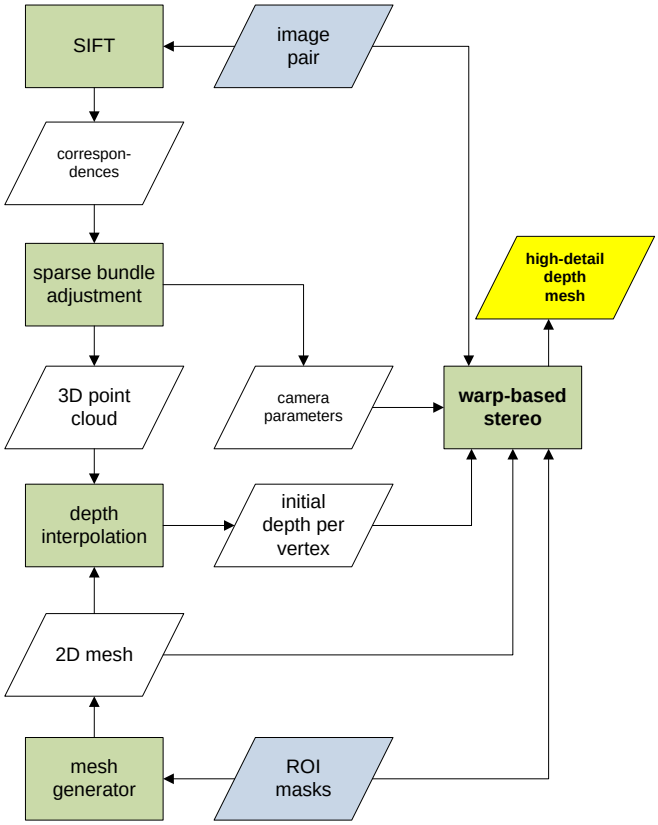


Figure 7: The data flow in the overall reconstruction system illustrates the role of initialization, calibration and mesh generation in the warp-based stereo system. Algorithms are shaded green; input to the system is shaded blue. Initial values for nonlinear optimization are computed by interpolating depth values of the dense reconstruction mesh from sparse interest point matches. For the walk-along stereo scenario, camera parameters are computed with sparse bundle adjustment. 2D meshing algorithms are used to generate the reconstruction mesh from mask images, indicating the region of interest to be reconstructed.

5.1. Initialization

Our reconstruction requires an initial camera calibration since the camera warp is not yet suited for calibration from scratch. For most images shown in the paper we used the Bundler software [29]. For the face and pose data-sets shown in some figures a calibration object was used. As stereo warp estimation is a highly non-linear problem we also need initial depth values. They are computed as follows: We first extract sparse feature correspondences (SIFT) and filter them according to consistency with epipolar geometry. Then we compute a depth value for each correspondence and find the initial depths for the vertices of the reconstruction mesh by scattered data interpolation with a Laplacian smoothness term. An example of an initialization is shown in figure 6.

5.2. Meshing

The mesh influences the quality of the reconstruction as well as the speed of the optimization: As the stereo warp function relies on the assumption of a piecewise planar 3D scene the density of the mesh determines how well smooth parts, fine details and depth discontinuities in the scene can be approximated. The density of the warping mesh also determines how the image data is partitioned for optimization, i.e. how many pixels contribute to the estimation of each vertex depth variable. If too many pixels contribute they are likely to drag the variable into many different directions in the optimization such that no meaningful descent direction can be found. If too few pixels contribute individual depth variables are likely to overfit on image noise and view-dependent lighting effects. Finally, the number of vertices in the mesh is equal to the number of variables to be estimated and thereby affects speed and memory useage. Vertex count also determines the size of the matrices used in the smoothness term, e.g. the mesh Laplacian. A good mesh is balances vertex density and pixel resolution for an optimal reconstruction result. Meshes were generated with the Constrained Delaunay algorithm of [21], which creates a triangulated mesh satisfying constraints on maximal triangle area while minimizing acute inner angles in triangles. We have conducted experiments with scene-adaptive meshes, where vertex density is higher in areas with high image detail. While this strategy allows to reduce vertex count in homogeneous regions, we found that it provides no qualitative advantage over dense but non-adaptive meshes in the highly detailed type of scenes shown throughout the paper.

5.3. Convergence

Convergence is very fast for well initialized regions with most surface details appearing in the first three iterations. In later iterations, sharpness at depth discontinuities improves and regions with worse initialization converge towards the correct result. This process is illustrated in figure 6. If the initialization is too far off the optimization cannot recover. In this case depth can be corrected approximately with the proposed tools before continuing the optimization. The results in the paper were computed in two passes with a mesh of 12 pixels per triangle on average in the first pass and 6 pixels per triangle in the second stage, which was initialized with the result of the first. Multi-resolution schemes are straightforward to use with our method, but we have yet to explore them systematically. We use a state-of-the sparse direct sparse solver (Intel Math Kernel Library DSS) for computing Gauss-Newton updates. Otherwise, our C++ implementation has not yet been systematically optimized. Average computation time was 2.3 sec per 10^6 vertices (i.e. variables) on a Xeon X5670. For the reconstruction in figure 1, with $3.5 \cdot 10^6$ triangles final resolution, this amounts to 8.1 sec per iteration in the second stage. With 9 iterations in the first and 5 in the second stage total reconstruction time was 93.6 sec. This timing is typical for the results shown, but there is significant room for optimization in the implementation. Mesh sizes are specified alongside the results in figure 9.

5.4. Results

Figure 9, as well as figures 1, 4 and 5 show results of the proposed method. It is important to note that, at the cost of higher iteration counts, we did *not* use our interactive tools to generate the results in these figures, except for the manual definition of ROIs. All were computed from two views. Instead of depth maps, we visualize them as shaded renderings of the reconstruction meshes used in the optimization to emphasize the surface details recovered by our algorithm. The challenging **museum artifact data-set** was captured to be representative for a "walk along" stereo application. The images were shot with a 12 megapixel consumer camera (Fuji X100) without tripod or flash, in dim lighting conditions (ISO 640-1200), and with a regular visitor's access to the objects. The stereo pairs were captured a few steps apart, and the object was kept in the image center which is more intuitive than trying to move a camera freehand along a horizontal baseline. The **actor poses and facial expressions** were captured at 18 megapixels (Canon EOS550D) with two synchronized fixed cameras and multi-flash lighting. We did not evaluate our method on Middlebury data due to its low image resolutions and the different scope of our algorithm. Also, the ground truth disparities in the Middlebury stereo data-sets are limited to pixel precision. Figure 9 G shows a reconstruction from frames 7 and 8 of the EPFL "Herz-Jesu-P25" data-set for multi-view algorithms. The images in this data-set are at the lower limit of image resolution for our method: While the nominal pixel resolution is 6 megapixels, the effective resolution is reduced by strong blur. Figure 9 E shows a shoe print reconstruction from a digital forensics project.

5.5. Limitations and outlook

A limitation of our approach is its dependence on initialization. The feature-based method works well for most scenes. However, feature mismatches occur even with epipolar geometry as filter, and the density and distribution of correspondences vary from scene to scene. Mismatches and feature sparsity lead to bad initializations from which the optimization either takes long to recover or does not recover at all. Our interactive tools provide means to correct these situations manually. Improving the robustness of initialization is part of our planned future work. We are currently evaluating classic binocular stereo methods on down-sampled images as an alternative to feature matching. Another limitation is the dependency on high image resolutions, as multiple pixels are subsumed under one mesh triangle. This does not scale well to low resolution content. Finally, occlusion handling is currently only implicit: The robust error norm in the data term reduces the influence of low confidence pixel matches and the smoothness terms propagate geometry into these regions. Established strategies such as backward matching (from $\mathcal{J}$ to $\mathcal{I}$, in our terminology) can be used to identify occlusions post-hoc. Note that backward matching is faster than forward matching with our method, as we can use the result of latter to initialize the former. We are also investigating the local weights of the norm-like functions as occlusion indicators.

5.6. Comparison

We have compared our approach to several algorithms: *123dCatch* by Autodesk, which offers 3D reconstruction as a web service; Patch-based Multi-view Stereo algorithm (*PMVS*) of [12], one of the most influential multi-view algorithms, with Poisson Surface Reconstruction [30] for meshing; *ScannerKiller*, a commercial binocular stereo application (time-limited demo). *123dCatch* offers different reconstruction resolutions of which we chose the highest for comparison. *ScannerKiller*'s options were set to highest quality. We have compared the algorithms on the museum artifact data-set which proved to be very challenging. Figure 8 shows detail crops from scenes in figure 1 and figure 8 A and I. Note that *123dCatch* requires at least three views, whereas our results and *ScannerKiller*'s are computed from two. To yield sufficient coverage, *PMVS* required 4 views on scene A, 2 views on the scene shown in figure 1 (3 views reproducibly fail) and 3 views for scene I. Despite of extensive parameter tuning we find it difficult to obtain fine geometric detail from PMVS on our data-set.

6. Conclusion

We have proposed a stereo algorithm which addresses high-detail surface reconstruction from binocular high-resolution input. Our method is capable of computing detailed geometry comparable to multi-view output from only two views. The algorithm is based on continuous optimization in a Gauss-Newton framework with robust cost functions in both data- and smoothness term. We have proposed a linearized stereo

Figure 8: Comparison on detail crops of the museum artifact data-set with other methods: *123dCatch* by Autodesk; *Patch-based Multi-view Stereo* (PMVS) of [12] with *Poisson Surface Reconstruction* [30] for meshing; *ScannerKiller*. For *123dCatch* and *ScannerKiller* the highest quality / resolution settings were used. The complete scenes are shown in figures 1 and 8 A and I. Note that *123dCatch* requires at least three views, whereas our results and *ScannerKiller*'s are computed from two. To yield sufficient coverage, *PMVS* required 4 views on scene A, 2 views on the scene shown in figure 1 (3 views reproducibly fail) and 3 views for scene I. Despite of extensive parameter tuning we find it difficult to obtain fine geometric detail from PMVS on our data-set. See also sect. 5

warp function which reduces the size of the optimization problem while using all pixels available. We have discussed several smoothness terms and proposed a robust mixed term which yields smooth geometry while allowing depth discontinuities. We have addressed methods for coping with photometric differences and proposed a computationally efficient photometric adaptation. We have shown that the continuous optimization approach enables extensions such as tools for a semi-interactive reconstruction work-flow, or warp-based camera refinement, which is straightforward to implement in the proposed framework.

Regarding future work, we plan to improve initialization and we want to model occlusions more explicitly. We also research further extensions along the line of the proposed camera calibration. For example, it is possible to parametrize the warp with a domain specific shape model instead of individual depths or camera parameters. For the interactive tools, user studies will help to quantify their benefits compared to traditional workflows and to optimize existing and new ways for users to interact with the reconstruction.

Appendix A. Robust Gauss-Newton

In the following the robust Gauss-Newton variant developed by Wedderburn and McCullagh [19, 20] is reviewed and motivated by comparison with classic Gauss-Newton. The classic Gauss-Newton algorithm is an optimization algorithm for a special class of energy functions, namely sum-of-squares functions:

$$\mathcal{E}(\mathbf{x}) = \sum_{i=1}^{n} r_i(\mathbf{x})^2 \tag{A.1}$$

The terms of the sum $r_i : \mathbb{R}^K \to \mathbb{R}$ are called residuals. With

$$\mathbf{r}(\mathbf{x}) = [r_1(\mathbf{x}) \dots r_n(\mathbf{x})]^T,$$

the gradient of $\mathcal{E}$ is,

$$\nabla\mathcal{E} = \sum_{i=1}^{n} r_i \cdot \nabla r_i = \mathbf{J}^T\mathbf{r} \tag{A.2}$$

where $\mathbf{J}$ is the Jacobian of $\mathbf{r}$. The Hessian of $\mathcal{E}$ is:

$$\nabla^2 e = \mathbf{J}^T\mathbf{J} + \sum_{i=1}^{n} r_i \cdot \nabla^2 r_i \tag{A.3}$$

The Gauss-Newton algorithm approximates the Hessian using only the left term of equation (A.3), i.e. $\nabla^2 e \approx \mathbf{J}^T\mathbf{J}$, when computing a Newton-type optimization step. The advantage of this approximation lies in the fact that no second derivatives are required to compute it. Substituting the approximation into the Newton update

$$\Delta_{\mathrm{N}} = -\left(\nabla^2\mathcal{E}\right)^{-1}\nabla\mathcal{E}$$

yields the Gauss-Newton step:

$$\Delta_{\mathrm{GN}} = -\left(\mathbf{J}^T\mathbf{J}\right)^{-1}\left(\mathbf{J}^T\mathbf{r}\right)$$

In equation (A.1), residuals with a large absolute value have a very large influence on the solution: As each residual is squared, the overall error is lowered the most by fitting the model mostly to these measurements. This may be problematic, as some these errors may be so large due to failed measurements or failure of the model in certain situations.

A robust version of equation (A.1) is obtained by replacing the square with a *norm-like function* $\rho : \mathbb{R} \to \mathbb{R}^{+0}$:

$$\mathcal{E}_{\mathrm{rob}}(\mathbf{x}) = \sum_{i=1}^{n} \rho(r_i(\mathbf{x}))$$

ρ is chosen to be less increasing than the square (at least for large values). Numerous robust cost functions have been proposed in the literature. The gradient of $\mathcal{E}$ is now:

$$\nabla\mathcal{E}_{\mathrm{rob}} = \sum_{i=1}^{n} \frac{\mathrm{d}\rho}{\mathrm{d}r_i} \cdot \frac{\mathrm{d}r_i}{\mathrm{d}\mathbf{x}} = \sum_{i=1}^{N} \frac{\mathrm{d}\rho}{\mathrm{d}r_i} \cdot \nabla r_i \tag{A.4}$$

For each norm-like function ρ the associated *weight function* is defined as:

$$\psi(r) := r^{-1}\frac{\mathrm{d}\rho}{\mathrm{d}r}$$

Substituting this to equation (A.4) yields:

$$\nabla\mathcal{E}_{\mathrm{rob}} = \sum_{i=1}^{n} r_i\psi(r_i)\nabla r_i$$

This reveals the similarity to the classic least squares gradient of equation (A.2) and can be rewritten analogously as

$$\nabla\mathcal{E}_{\mathrm{rob}} = \mathbf{J}^T\mathbf{W}\mathbf{r}$$

with

$$\mathbf{W} := \mathrm{diag}(\psi(r_1) \dots \psi(r_n)).$$

$\mathbf{J}$ is the Jacobian of the residual vector $\mathbf{r}$, as above.

The Hessian of e_{rb} is:

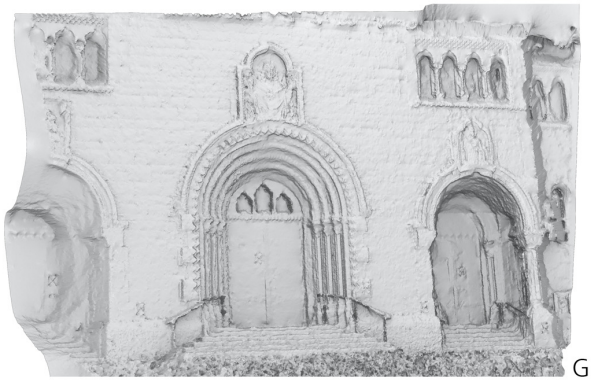
$$\nabla\left(\nabla\mathcal{E}_{\mathrm{rob}}^T\right) = \sum_{i=1}^{n}\left(\nabla r_i\psi(r_i)\nabla r_i^T + r_i\frac{\partial\psi}{\partial r_i}\nabla r_i\nabla r_i^T + r_i\psi(r_i)\nabla^2 r_i\right).$$

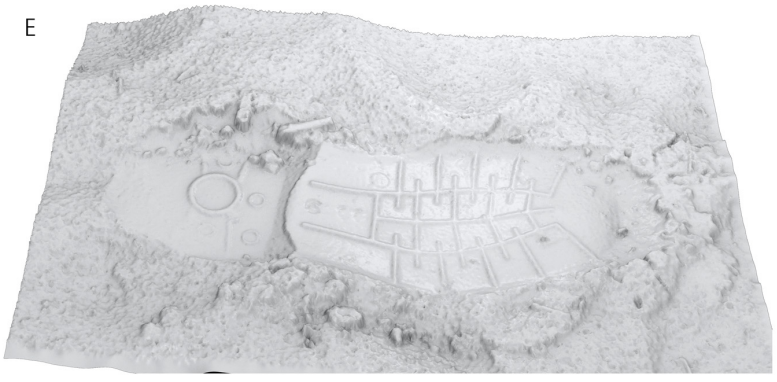
As in the classic Gauss-Newton algorithm for least squares, the Hessian is approximated by omitting the higher order terms. The remaining terms, written in matrix form, is again similar to the classic least squares case:

$$\nabla\left(\nabla\mathcal{E}_{\mathrm{rob}}^T\right) \approx \sum_i\left(\nabla r_i\psi(r_i)\nabla r_i^T\right) = \mathbf{J}^T\mathbf{W}\mathbf{J}$$

With the gradient and approximate Hessian defined, the optimization can be performed exactly as in the least squares case. The weight function ψ has to evaluated in every iteration in order to build up $\mathbf{W}$.

Figure 9: Results computed with the proposed method, shown as shaded meshes rather than depth maps to emphasize surface detail. The meshes are the ones used in the optimization. All results were computed from two images. Scenes A, D, F, H and I and figure 1 were captured freehand in a museum under available light. B and C as well as figures 6 and 4 were captured with synchronized cameras and flash lighting. G is based on frames 7 and 8 of the EPFL "Herz-Jesu-P25" data. E is a shoe print reconstructed for a digital forensics application. Vertex counts are given in the table above. Data-sets are described in section 5.





Number of triangles used in the different results:

scene	A	B	C	D	E	F	G	H	I	J	fig. 1
vertex count·10^3	336	353	376	157	120	151	487	318	251	158	352

References

[1] Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision 2002;47:7–42. doi:10.1023/A:1014573219977.

[2] Hirschmuller H, Scharstein D. Evaluation of cost functions for stereo matching. In: Proc. IEEE CVPR. 2007,doi:10.1109/CVPR.2007.383248.

[3] Bleyer M, Rother C, Kohli P. Surface stereo with soft segmentation. In: Proc. IEEE CVPR. 2010,.

[4] Yang Q, Wang L, Yang R, Stewenius H, Nister D. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. Trans on Pattern Analysis and Machine Intelligence 2009;31(3):492 –504. doi:10.1109/TPAMI.2008.99.

[5] Klaus A, Sormann M, Karner K. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In: Proc. IEEE CVPR. 2006,doi:10.1109/ICPR.2006.1033.

[6] Bleyer M, Rother C, Kohli P, Scharstein D, Sinha S. Object stereo — joint stereo matching and object segmentation. In: Proc. IEEE CVPR. 2011,.

[7] Campbell N, Vogiatzis G, Hernandez C, Cipolla R. Using multiple hypotheses to improve depth-maps for multi-view stereo. In: Proc. IEEE ECCV. 2008,.

[8] Yang Q, Yang R, Davis J, Nister D. Spatial-depth super resolution for range images. In: Proc. IEEE CVPR. 2007,.

[9] Bradley D, Boubekeur T, Heidrich W. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In: Proc. IEEE CVPR. 2008,.

[10] Beeler T, Bickel B, Beardsley P, Sumner B, Gross M. High-Quality Single-Shot Capture of Facial Geometry. ACM Transactions on Graphics 2010;29(3).

[11] Goesele M, Curless B, Seitz SM. Multi-view stereo revisited. In: Proc. IEEE CVPR. 2006,.

[12] Furukawa Y, Ponce J. Accurate, Dense, and Robust Multi-View Stereopsis. Trans on Pattern Analysis and Machine Intelligence 2008;1:1–14. doi:10.1109/TPAMI.2009.161.

[13] Gargallo P, Prados E, Sturm P. Minimizing the reprojection error in surface reconstruction from images. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. 2007, p. 1–8.

[14] Delaunoy A, Prados E, Gargallo P, Pons JP, Sturm P. Minimizing the Multi-view Stereo Reprojection Error for Triangular Surface Meshes. In: BMVC. Leed, UK; 2008,.

[15] Lowe DG. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 2004;60, 2:91–110. URL: `http://www.cs.ubc.ca/ lowe/keypoints/`.

[16] Habbecke M, Kobbelt L. An intuitive interface for interactive high quality image based modelling. In: Pacific Graphics. 2009,.

[17] Baker S, Matthews I. Lucas-Kanade 20 Years On: A Unifying Framework. International Journal of Computer Vision 2004;56(3):221 –55.

[18] Huber P. Robust Statistics. John Wiley & Sons; 1981.

[19] Wedderburn RWM. Quasi-likelihood functions, generalized linear models, and the Gauss-Newton method. Biometrika 1974;61(3):439–47.

[20] McCullagh P, Nelder J. Generalized Linear Models. Chapman & Hall; 1998.

[21] Shewchuk JR. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In: Lin MC, Manocha D, editors. Applied Computational Geometry: Towards Geometric Engineering; vol. 1148. Springer; 1996, p. 203–22.

[22] Hirschmuller H. Accurate and efficient stereo processing by semi-global matching and mutual information. In: Proc. IEEE CVPR. 2005,.

[23] Mei X, Sun X, Zhou M, Jiao S, Wang H, Zhang X. On building an accurate stereo matching system on graphics hardware. In: Proc. IEEE ICCV Workshops. 2011,.

[24] Pons JP, Keriven R, Faugeras O. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. Int J Comput Vision 2007;72(2):179–93. URL: `http://dx.doi.org/10.1007/s11263-006-8671-5`. doi:10.1007/s11263-006-8671-5.

[25] Wu C, Wilburn B, Matsushita Y, Theobalt C. High-quality shape from multi-view stereo and shading under general illumination. In: Proc. IEEE CVPR. 2011,.

[26] Hilsmann A, Eisert P. Joint Estimation of Deformable Motion and Photometric Parameters in Single View Video. In: ICCV Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment. Kyoto; 2009,.

[27] He K, Sun J, , Tang X. Guided image filtering. In: Proc. IEEE ECCV. 2010,.

[28] Lourakis MA, Argyros A. SBA: A Software Package for Generic Sparse Bundle Adjustment. ACM Trans Math Software 2009;36(1):1–30. doi:http://doi.acm.org/10.1145/1486525.1486527.

[29] Snavely N, Seitz SM, Szeliski R. Photo tourism: Exploring photo collections in 3D. ACM Trans Graph 2006;25(3):835–46. doi:10.1145/1141911.1141964.

[30] Kazhdan M, Bolitho M, Hoppe H. Poisson surface reconstruction. In: Proc. ESGP. ISBN 3-905673-36-3; 2006,.