Multiview Normal Field Integration using Graph-Cuts

Aljosa Osep*
Supervised by: Michael Weinmann,† Reinhard Klein‡

Institute of Computer Science II - Computer Graphics
University of Bonn
Bonn / Germany

Abstract

While there are many algorithms which address the reconstruction of partial surfaces from single-view normal fields, to the best of our knowledge, there is only one method [Chang et al., 'Multiview Normal Field Integration using Level Sets', CVPR 2007] which focuses on the reconstruction of the full shape of an object from multiple normal fields captured from multiple viewpoints. In this paper, we propose an alternative approach for the integration of such normal fields. We use a similar energy formulation as Chang et al., but replace the employed level-set representation with a graph-cut based reconstruction. Based on the visual hull of the object we estimate the visibility from each viewpoint. Then we project estimated normal fields to the visual hull and the optimal surface is computed by maximizing the flux of the obtained vector field through the surface. The graph-cut approach allows for a fast and globally consistent optimization of the given problem. Finally, we demonstrate the validity of our algorithm on synthetic data sets.

Keywords: 3D reconstruction, normal field integration, graph-cuts, multiview vision

1 Introduction

The digitization of 3D objects is a very important topic in computer graphics and computer vision and requires a faithful reconstruction of the object surface. There are various applications to surface reconstruction of real models in industry, archaeology and art, medical imaging and entertainment. The aim of surface reconstruction is to construct an as accurate as possible approximation of the geometry of a real surface.

For this purpose, many different techniques have been developed. Some methods, such as laser scanning, structured light systems and multiview stereo methods directly reconstruct an oriented point cloud, from which a closed surface can be derived by applying one of the surface fitting methods [13, 4, 20, 18, 12]. In contrast, there are also approaches that only rely on information about the surface normals. Methods such as Shape-from-Shading [23] and Shape-from-Specularity [6] can be used for obtaining a normal field and subsequently, normal field integration methods are applied for estimating the shape of the observed object. However, most of the works address only reconstruction of a certain part of the object surface from the estimated normal fields as they only use a single viewpoint. In this paper, we address the integration of such estimated normal fields from multiple views in order to recover the full 3D shape of the object. The fusion of such information from multiple viewpoints is a challenging problem, because we do not have any spatial surface samples but only normal samples.

To the best of our knowledge, this problem has only been addressed in [5]. The authors derive an energy functional consisting of a surface and the flux term, and minimize it using level-sets. Our work is based on the minimization of the same energy functional. However, we take a different optimization approach. Similarly as in [5], we first compute the visual hull of the object. Based on that we compute an approximation of the visibility. Then, we project the observed normal fields from each view to the visual hull, taking into the account the visibility. We simultaneously optimize the flux through the surface by maximizing divergence and enforce a minimal surface using Graph-Cuts.

The rest of our paper is organized as follows. In the next section, we discuss previous and related work in the area of 3D surface reconstruction using normal field integration. In Section 3, we state our goal more precisely and introduce necessary notations. In Section 4, we explain all steps of our algorithm in detail and in Section 5 we discuss the results of our multiview normal field integration algorithm. Finally, we conclude the paper in Section 6 and discuss possibilities for future work.

2 Previous Work

As pointed out in [11], there is no algorithm, which is able to reconstruct a general scene with any type of materials and lighting conditions. The different approaches are usually designed according to the requirements in the

*osep@informatik.uni-bonn.de
†mw@cs.uni-bonn.de
‡rk@cs.uni-bonn.de

observed scene. For this, they rely on exploiting different visual cues such as silhouettes, textures, shading, specularities, etc.

Assuming we already have normal information as input, we concentrate on reviewing techniques using such normal information to reconstruct a surface. There exists a variety of techniques addressing the normal field integration problem. This is a challenging problem, because in the presence of noise, the observed normal fields are not integrable. For a vector field to be gradient of the function, its curl must be zero, which is rarely the case in the presence of noise and outliers.

Most of the methods addressing this issue attempt to enforce integrability. In [8], the authors project the gradient field to integrable functions using the Fourier basis functions. There also exist several variants of this method, where different basis functions (cosine basis [9], shapelets [16], etc.) are used. Another approach, proposed in [22], attempts to solve the problem by finding the function whose gradient is closest to the observed normal field in the L_2 norm sense by solving the Poisson equation. The authors of [21] observed, that methods based on minimizing least-squares cost functions cannot handle outliers well and propose a method that minimizes an energy functional in the L_1 norm sense. However, all of these methods only address the reconstruction of partial (2.5D) surfaces from a single-view normal field. A detailed review of the related work on single-view normal field integration can be found in [23].

Combining normal field information from several views is addressed in [5]. There, the authors propose an efficient algorithm based on energy minimization, to which our approach is closely related. To the best of our knowledge, their algorithm is the first one that is able to recover the full 3D shape of an object from multiple normal fields. In their work, they derive a geometric PDE that minimizes an energy functional which is composed of mean curvature and flux term. The PDE is optimized using a level set method. This optimization process is a drawback of their approach, since it can find only a local minimum of the energy functional and it is highly dependent on the initial surface.

There are also certain similarities of the considered reconstruction problem to surface reconstruction from oriented point clouds. Among the implicit function fitting-based methods, there are approaches [13, 4, 18] that consider an oriented point cloud simply as a vector field that is in fact a sparse sampling of a continuous vector field corresponding to the true surface ∂M of the solid M. Based on that observation, these methods attempt to find an implicit function $f(x)$ whose gradient $\nabla f(x)$ is as close to the observed vector field $\vec{V}$ as possible. Our reconstruction method is also based on these ideas and observations.

3 Problem Statement

Our goal is to reconstruct the full 3D, closed and twice differentiable surface $\partial M \in \mathbb{R}^3$ of a solid M, given the observed normal fields (normal samples) from different cameras. Furthermore, we want to obtain a watertight surface in form of a polygonal mesh. Note, that in our formulation, we do not have samples of surface points but only samples of their normals, projected to the image planes of the cameras. In our setup, we assume having N calibrated cameras $C_i, i \in [1...N]$, as visualized in Figure 1. Each camera provides a noisy normal field estimate v_i. The mapping $v_i : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{R}^3$ is a projection of normals of the points, seen from the camera C_i to its image plane. For the cameras, we assume a pinhole camera model and we assume that we have for each camera a projection matrix $P_i = K_i[R_i|t_i]$. The matrix K_i is the camera calibration matrix and it provides the intrinsic parameters of the camera. The matrix $[R_i|t_i]$ provides information about external camera parameters (position and orientation in space). The projection of the point $x \in \partial M$ to the image plane of i-th camera C_i will be denoted by $\widetilde{x_i} = P_i x$ and $v_i(\widetilde{x_i})$ denotes the projection of the normal $n(x)$ of the point $x \in \partial M$ to the image plane of camera C_i.

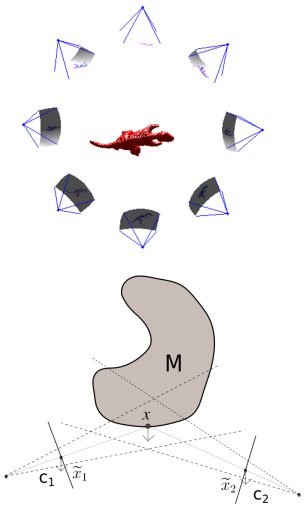

Figure 1: A typical scene setup *(top)*. Projection process *(bottom)*. The surface point x of the solid M is visible in cameras C_1 and C_2 and its normal is projected to their image planes.

We consider a very similar projection process as the authors of [5], which is illustrated in Figure 1. Taking into account the projection process, additive noise and visibility constraints, our normal field formation model can be expressed by:

$$v_i(\widetilde{x_i}) = n(x) + \eta_i, \quad \forall x \in \left\{ x | \psi_{i,\partial M}(x) = 1 \right\} \qquad (1)$$

where $n(x)$ is normal of the point x, η_i is the corresponding

1. Compute the vector field $\vec{V}$
   (a) Initialize the model by silhouette carving
   (b) Compute the visibility
   (c) Project normal fields
2. Compute the divergence of $\vec{V}$
3. Construct a graph
   (a) Establish *n-links* (Adjacent nodes)
   (b) Establish *t-links* (Terminals)
4. Compute the Min-Cut on the constructed graph using the TouchExpand algorithm [18]
5. Extract the isosurface using Marching Cubes

Table 1: The main steps of our algorithm.

additive noise and

$$\psi_{i,\partial M}(x) \doteq \begin{cases} 1, & x \text{ is visible from the camera } C_i \\ 0, & \text{else} \end{cases} \quad (2)$$

is a visibility function that indicates whether the point x is visible from the camera C_i. Like in [5], we also address inferring the true coordinates of all points $x \in \partial M$ from estimated noisy normal fields and reconstructing the full 3D solid M as accurate as possible.

4 Proposed Algorithm

The problem domain is discretized on a regular grid. Then, the vector field $\vec{V}$ is computed and the energy functional is transformed to a graph. Subsequently, we compute the surface ∂M as a Min-Cut on the energy graph. The polygonal mesh is obtained using the Marching Cubes algorithm [19]. The main steps of our algorithm are listed in Table 1 and detailed explanations of all steps are provided in the following subsections.

4.1 Energy Minimization Framework

Energy minimization is a very popular approach and often used for surface reconstruction, both in the field of single-view normal field integration [23] and surface reconstruction from point clouds [13, 4, 18]. In the first paper (to the best of our knowledge) addressing the problem of multi-view normal integration [5], the problem is formulated in terms of energy minimization as well. Our formulation is based on the very similar energy functional.

The authors of [5] are motivated by the approach described in [22], where a relief surface is reconstructed from a single normal field in the following way:

$$E(Z) \doteq \int_Z (\nabla x - v(\widetilde{x}))dA \quad (3)$$

where the integral is over the planar (image) domain Z. Intuitively, the energy functional penalizes discrepancy between the gradient of the surface we would like to obtain and the observed normal field. The authors of [5] extend the idea to the domain of the surface ∂M. Taking into account the visibility constraints, their energy functional is as follows:

$$E(\partial M) \doteq \int_{\partial M} \frac{1}{N_{\partial M}(x)} \sum_{i=1}^{N} \psi_{i,\partial M}(x) \, \|n(x) - v_i(\widetilde{x_i})\|^2 \, \mathrm{dA} \quad (4)$$

where the term

$$N_{\partial M}(x) \doteq \sum_{i=1}^{N} \psi_{i,\partial M}(x) \quad (5)$$

denotes the number of cameras in which the surface point x was seen. The proposed energy functional minimizes differences between the normals $n(x)$ of the surface points $x \in \partial M$ and all observed normal samples, from all cameras. The normal sample of the point x, seen by the camera C_i, corresponds to the normal of the projection of x to the i-th image plane. Additionally, for each camera, we take into account only the visible points and we normalize the total deviation by the number of cameras the point was seen from.

Alternatively, we are again looking for the surface whose gradients match best to the observed normals in a least squares sense under the visibility constraints. The authors of [5] observed, that minimizing Equation (4) is equivalent to maximizing the flux through the surface and simultaneously minimizing the surface area. Hence, we obtain a similar energy functional as proposed in [18], which consists of a data term and a regularization term:

$$E(\partial M) = \lambda_1 R(\partial M) - \lambda_2 D(\partial M). \quad (6)$$

Here,

$$R(\partial M) = \int_{\partial M} \mathrm{dA} \quad (7)$$

denotes the area over the surface, and

$$D(\partial M) = \int_{\partial M} n(x)\vec{V}(x)\mathrm{dA}, \quad (8)$$

describes the flux through the surface ∂M. Considering the visibility from each camera, the vector field $\vec{V}$ can be computed according to

$$\vec{V}(x) \doteq \frac{\sum_i \psi_{i,\partial M}(x) v_i(\widetilde{x})}{N_{\partial M}(x)}. \quad (9)$$

The implementation of the vector field computation will be discussed in the next section.

Additionally, we weight both terms. Intuitively, by minimizing this energy functional, we are aligning the surface ∂M with the vector field $\vec{V}$, and with the regularization term, we avoid over-fitting and make the optimization process robust to noise and outliers.

4.2 Computation of the Vector Field

The computation of the vector field $\vec{V}$, as stated in Equation (4), requires summing over all observed normal fields while accounting for visibility and occlusion. That is a non-trivial task, since we do not know where the true surface ∂M lies, i.e. we do not have any information about the real position of the surface points. We approach the

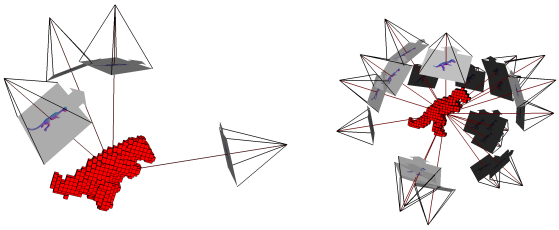

Figure 2: A silhouette carving process. Result of carving using 4 cameras *(top left)* and result using 20 cameras *(top right)*.

problem by exploiting an additional visual cue about the geometry of our solid M that the estimated normal fields provide. To be exact, we compute the visual hull of the object based on the silhouettes obtained from the normal fields [17]. The silhouette carving process is visualized in Figure 2.

We discretize our domain of interest using a regular grid. For this, we first extract the area of interest by intersecting the volumes of all cameras. Then we narrow down this area of interest by performing an initial carving. Doing so, we obtain a rough carved object. We take the maximal and minimal point of this object and we use them to initialize the final bounding volume, on which we perform a fine carving.

Based on the visual hull, which is computed on the initialized voxel grid, an approximation of the visibility function can be computed. Note, that in order to compute the exact visibility function and account for all occlusions precisely, we would need the actual surface ∂M, which is just what we want to obtain. The approximate visibility for the camera C_i is computed by casting rays from the focal point of C_i to the centres of all voxels in the voxel grid. The visibility is determined by intersecting each ray with a triangular mesh representing the visual hull, obtained by Marching Cubes [19]. The location of the exact surface ∂M is not known at this stage, but we know that it must be somewhere close to the visual hull. Furthermore, we know that the actual surface can only be smaller than the visual hull. For that reason, we do not only consider border voxels as visible, but we rather take into account visible bands of voxels, as shown in Figures 3 and 4. A visible band consists of the border voxels, directly visible from camera C_i and the voxels that are for a band-depth ε away from the border voxels towards inner region of the model in the direction of the rays. The effects of the parameter ε will be discussed in Section 5.

Figure 3: Slice of the computed visibility on the voxel grid from the i-th camera. Dark (blue) voxels correspond to the visual hull and bright (green) voxels are visible from camera C_i.

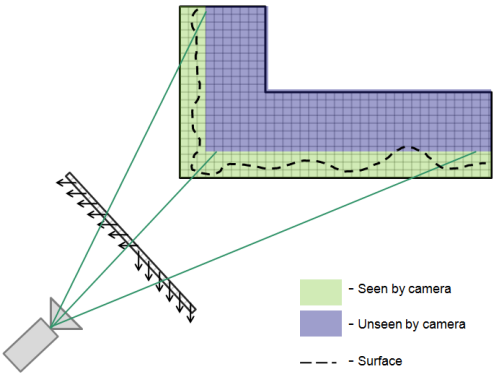

Figure 4: Normal projection from the image plane to the visible band *(green)*. Note, that the surface ∂M is unknown, but we estimate, that it lies within the visible band.

At this point, the computation of the vector field $\vec{V}$ is simple. From each camera, we project the observed normal fields to the visible band of the voxels, as demonstrated in Figure 4. This is efficiently computed using the back-projection of voxels to the image planes. Note, that in case of silhouette carving [17], each voxel is back-projected to the image plane of each camera. Based on a lookup to the silhouette map, a voxel is either kept or rejected, depending on silhouette consistency. In the normal projection case, for each camera, we project all voxels that are in the visible band to the image plane and make a lookup to the color-coded normal map, which is visualized in Figure 5. The obtained normal is then assigned to the voxel being projected. The value at each voxel is divided by the total number of cameras the voxel was seen from. As a consequence of this step, we obtain the vector field $\vec{V}$, which is discretized on a regular grid. Alternatively, this projection process can be seen as normal voting. We assign votes to voxels near the visual hull for being real

Figure 5: A color-coded normal field observed from the i-th camera and the corresponding silhouette.

surface voxels. The next step is the optimization of the flux through the surface represented by the vector field $\vec{V}$ using a Graph-Cut technique, which is described in the next section.

4.3 Energy Optimization using Graph-Cuts

Graph-Cuts have been successfully applied to various computer vision problems such as image restoration [3], stereo vision [3] and segmentation [14]. In [15], it is shown, that it is possible to globally optimize a wide class of geometrically motivated hypersurface functionals with Graph-Cuts. Specifically, it is shown that that any functional that consists of a combination of area/length and flux of the given vector field can be optimized by Graph-Cuts, which is just what we need in our case. Motivated by the successful application of Graph-Cuts to the problem of reconstructing a surface from oriented point clouds [18], we decided to optimize our energy functional (6), also consisting of area and flux term, via Graph-Cuts.

As pointed out in [18], maximizing the flux term of the energy functional (8) is equivalent to optimizing the divergence of the vector field $\vec{V}$ in the interior (Gauss-Ostrogradsky a.k.a. Divergence theorem). Thus, the final energy we are optimizing is:

$$E(\partial M) = \lambda_1 \int_{\partial M} \mathrm{dA} - \lambda_2 \int_M \mathrm{div}(\vec{V}(x))\mathrm{dV}. \qquad (10)$$

This energy can be simply converted into a graph. Adjacent nodes, which are corresponding to voxels in a voxel grid, are connected by *n-links*, and additionally, the nodes are connected to terminals s and t, based on the divergence of $\vec{V}$ (see Figure 6). In the constructed graph, *t-links* are weighted proportional to the absolute value of the divergence at each voxel. Furthermore, voxels with positive volumetric potentials are connected to the source node and voxels with negative volumetric potential are connected to the terminal node. To minimize metrification artifacts, the voxel nodes are linked not only to directly adjacent nodes, but to larger neighbourhoods. The size of the actual neighbourhood can be considered as a parameter. In most of our experiments, we considered neighbourhoods of 26 nodes. The weights of the *n-links* are proportional to the weight λ_1. For detailed information on the weight computation, we refer to [1]. The solution can then be obtained by computing an *s/t-cut* on the graph. In a sense, we are solving

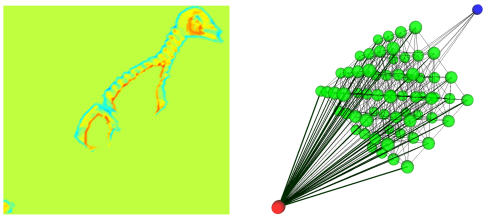

Figure 6: Slice of the computed divergence on the voxel grid from *(left)*. Simple graph constructed on a voxel grid *(right)*. Green nodes correspond to the voxels, red *(leftmost)* node and blue *(rightmost)* node represent the terminals.

a segmentation problem: we are segmenting voxels that belong to the model M from the background.

There exists a variety of efficient algorithms for solving the Min-Cut/Max-Flow problem, starting with Ford-Fulkerson [7] and "push-relabel" [10]. In [2], an exhaustive overview of Min-Cut/Max-Flow algorithms is given, focusing on computer vision problems. The authors of [18] observed, that due to high-resolution grid demands for surface reconstruction, the use of existing Min-Cut/Max-Flow algorithms on a regular grid is practically infeasible and proposed a novel TouchExpand algorithm. Their algorithm computes a global cut while it keeps in memory only local graphs. Because of our regular-grid based discretization, the TouchExpand is method of choice for our surface reconstruction purpose, although it would be an even better idea to use an adaptive grid and thus reduce size of the graph drastically. This way, use of algorithms discussed in [2] would be feasible.

5 Results

To validate our approach, we conducted several experiments. As we do not address the estimation of normal fields but instead assume having such information, we evaluate our approach only on synthetic data, generated with our testing environment, described in Section 5.1. In order to test the robustness of our algorithm, we also added noise to our data sets. We performed tests on the Utah Teapot and Cyberware Dinosaur model (Figure 7).

We implemented our algorithm in Matlab and partially, in C++, and executed it on a PC with a Core2Duo 6600 CPU (2.4GHz) processor and 4GB RAM. While the optimization process takes less than one minute (30 seconds on average), the naive visibility computation can take up to several hours, depending on the number of cameras. In case of 16 cameras, it takes about 1.5 hours.

Because of our Matlab implementation, we had to limit our grid size. The actual resolution depends on the dimensions of the model.

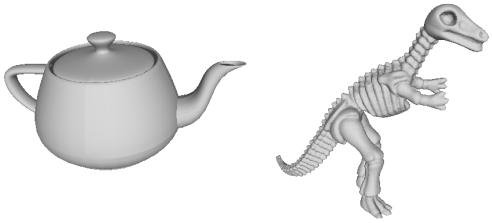

Figure 7: The ground truth: Utah Teapot model *(left)* and Cyberware Dinosaur model *(right)*.

5.1 Testing Environment

We generated our synthetic data sets with a testing environment we developed for the evaluation of our algorithm. The testing application allows the user to place a 3D model into a scene and virtual cameras around it. From these virtual cameras, ground-truth normal images are created by colouring pixels according to their true normals using a simple pixel shader program. In addition, the testing environment is also capable of simulating noisy measurements by adding Gaussian or salt&pepper noise to the rendered normal images.

The generation of synthetic normal fields is not the only purpose of our testing application. We are also planning to make use of it for testing out different techniques for normal estimation of different materials (highly-specular, glass, etc.). Different surface reflectance behaviours can be simply simulated by selecting a different shader program.

5.2 Results on Synthetic Data

In our initial experiments, we used a virtual setup where the object is surrounded by several cameras, as shown on Figure 1. The results using different numbers of cameras and depth parameters ε are visualized in Figures 8, 10, 9 and 11. We observe, that our algorithm is able to produce reasonable reconstructions even when using only a few cameras. Using normal information, also concave regions can be recovered as shown in the Dinosaur example on Figures 8, 9 and 10.

However, the algorithm at this stage is somehow sensitive to the choice of the parameter ε. For thinner bands ($\varepsilon = 3$), it may happen that normals are not projected to the voxels where the surface ∂M actually is located. The effect is visible at the eye of the Cyberware Dinosaur's head in Figure 10. Although the concave region around the eye is recovered to some degree, the algorithm clearly does not reach the desired surface ∂M in this region. In case of choosing deeper bands ($\varepsilon = 6$), the concave region around eye is reconstructed well. For visibility bands of arbitrary depth, reconstruction artifacts (marked with the blue circle) may occur. The reason for this is that in some regions, normal votes from different sides may interfere with each other. How to solve problems related to visibility-band depth parameter will be discussed in Section 6.

Just as in the algorithm presented in [5], our algorithm is also dependent on the initial surface. How to resolve that issue, is also addressed in Section 6. In fact, we believe that with very simple improvements, we will be able to completely skip the silhouette carving step.

In case of Utah Teapot dataset we observe, that the Graph-Cut was able to deal easily with the large uncarved area in the bottom of the Teapot. That region did not receive any normal votes, so the Graph-Cut simply produced the minimal surface at the bottom of the cup. Overall we obtain a nice reconstruction of the Teacup, although we can observe discretization artifacts due to the limited voxel grid size.

In a real-world situation, due to the acquisition process, normal estimates will always contain noise, this is why we corrupted our perfect normal maps with Gaussian and salt&pepper noise. It should also be noted, that normal estimates may be erroneous and that normal fields will not neccessarily match on the real input data. There are also other possible sources of errors that we do not consider here, for example, systematic error due to imprecise camera calibration, errors caused by lighting conditions, outliers, etc. In Figure 12 we can observe, that in presence of light salt&pepper noise and Gaussian noise with standard deviation $\sigma = 0.05$, reconstruction results are not significantly affected. In case of strong Gaussian noise ($\sigma = 0.2$), we observe reconstruction artifacts, but still the algorithm is able to recover the rough shape. From these tests we conclude, that our algorithm is reasonably robust to the noise.

6 Conclusions and Future Work

In this paper, we propose an approach to solve the multiview normal integration problem via Graph-Cuts. For the discretized version of the utilized energy functional under the given approximation of the visibility, our approach produces a globally optimal solution. Furthermore, the energy optimization with Graph-Cuts using the Touch-Expand algorithm is very efficient. The bottle-neck of our approach is the visibility computation, which can be easily optimized and remains our future work. We show, that our algorithm performs reasonably well even on challenging models like the Cyberware's Dinosaur and is very robust to the noise.

One drawback of our approach is its sensitivity to the band parameter and dependence on the visual hull. To resolve this issue, we are planning to implement an iterative approach. In that case, the normals will be projected to the model reconstruction from the last iteration, M_{k-1}, using a very thin visibility band. At each iteration, the visibility will be computed and the Graph-Cut will be applied again to iteratively refine the reconstruction. Such an iterative approach also allows to skip the silhouette carving process. Further improvements should also consider a higher

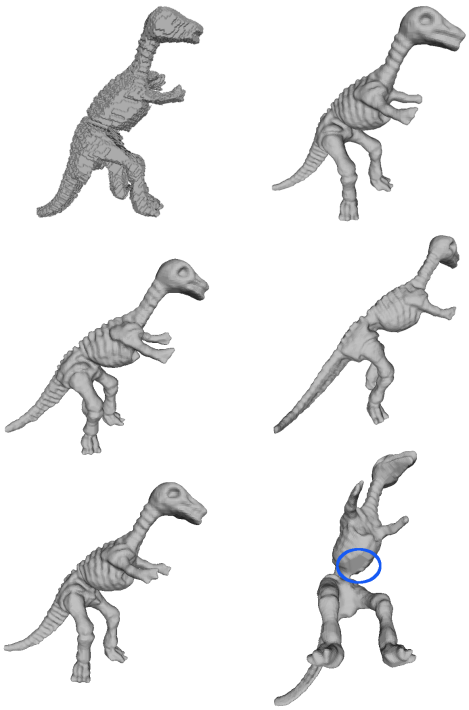

Figure 8: Reconstructions of the Cyberware Dinosaur datasets with band parameter $\varepsilon = 6$. First column: visual hull, reconstruction using 6 cameras and reconstruction using 16 cameras. Second column: reconstructions using 16 cameras. The blue circle marks area that was not seen by the cameras.

grid resolution in order to achieve more accurate reconstruction results.

We also observe, that it is more likely that the surface ∂M coincides with voxels, where projected normals i.e. normal votes match. For that reason, we are planning to incorporate an additional energy functional term, that penalizes differences between the normal votes and the mean normal at each voxel.

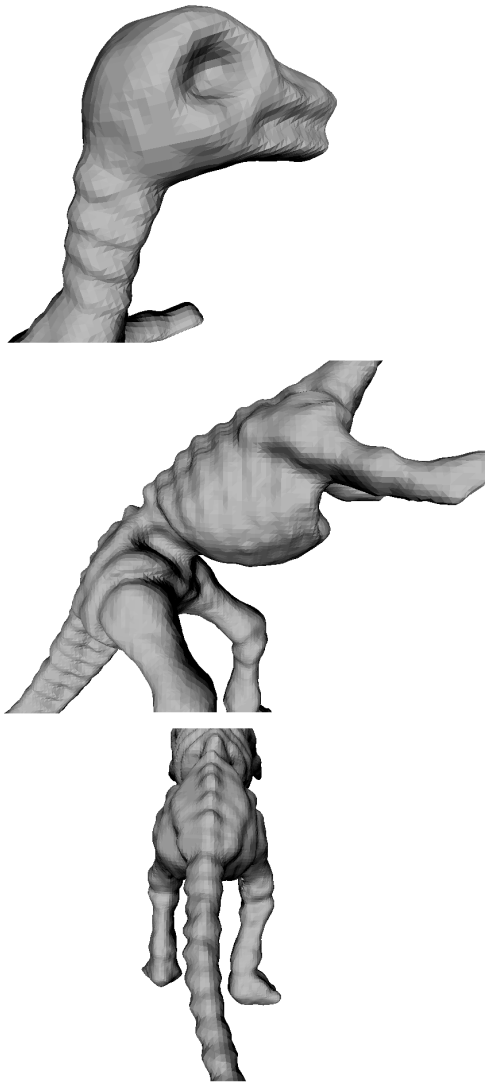

Figure 9: Details of the Cyberware Dinosaur's reconstruction using 16 cameras and $\varepsilon = 6$.

References

[1] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. ICCV '03, 2003.

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI*, 26:1124–1137, September 2004.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23:1222–1239, November 2001.

[4] F. Calakli and G. Taubin. Ssd: Smooth signed distance surface reconstruction. *Comput. Graph. Forum*, 30(7):1993–2002, 2011.

[5] J. Y. Chang, K. M. Lee, and S. U. Lee. Multiview normal field integration using level set methods. In *CVPR'07*, 2007.

[6] T. Chen, M. Goesele, and H. Seidel. Mesostructure from specularity. In *CVPR (2)*, pages 1825–1832, 2006.

[7] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[8] R. T. Frankot and R. Chellappa. A method for enforcing integrability in shape from shading algorithms. *IEEE TPAMI*, 10:439–451, July 1988.

[9] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE TPAMI*, 23:643–660, June 2001.

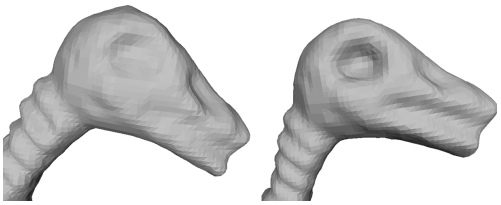

Figure 10: Cyberware Dinosaur's head reconstructed using 16 cameras with $\varepsilon = 3$ *(left)* and $\varepsilon = 6$ *(right)*

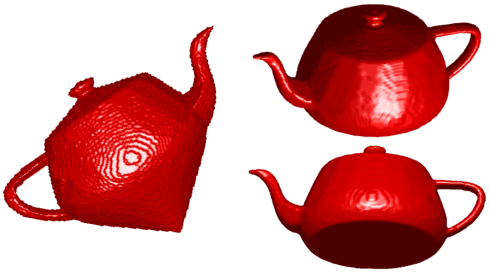

Figure 11: Reconstructions of the Utah Teapot: visual hull *(left)*, result after Graph-Cut *(right)*.

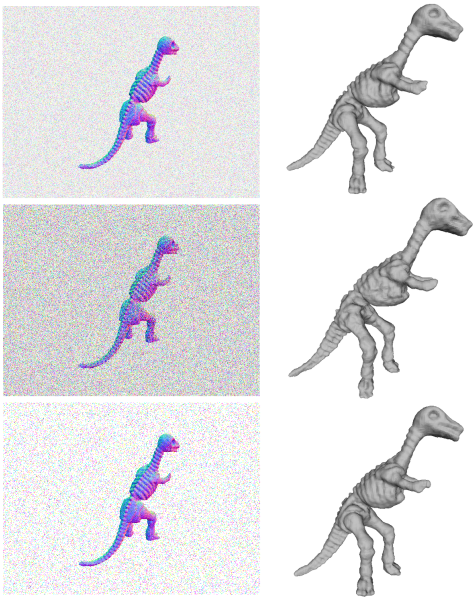

Figure 12: Reconstructions of noisy Cyberware Dinosaur data sets. In first two reconstructions, Gaussian noise with standard deviations $\sigma = 0.05$ and $\sigma = 0.2$ was added. Third reconstruction shows result for salt&pepper noise.

[10] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. STOC '86, pages 136–146, 1986.

[11] C. Hernández and G. Vogiatzis. Shape from photographs: A multi-view stereo pipeline. In Roberto Cipolla, Sebastiano Battiato, and Giovanni Maria Farinella, editors, *Computer Vision: Detection, Recognition and Reconstruction*, volume 285 of *Studies in Computational Intelligence*, pages 281–311. Springer, 2010.

[12] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. SIGGRAPH '92, pages 71–78, 1992.

[13] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, 2006.

[14] J. Kim, J. W. Fisher, III, A. Tsai, C. Wible, A. S. Willsky, and W. M. Wells, III. Incorporating spatial priors into an information theoretic approach for fmri data analysis. MICCAI '00, pages 62–71, 2000.

[15] V. Kolmogorov and Y. Boykov. What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. ICCV '05, pages 564–571, 2005.

[16] P. Kovesi. Shapelets correlated with surface normals produce surfaces. ICCV '05, pages 994–1001, Washington, DC, USA, 2005. IEEE Computer Society.

[17] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE TPAMI*, 16:150–162, February 1994.

[18] V. Lempitsky and Y. Boykov. Global optimization for shape fitting. In *CVPR, 6*, 2007.

[19] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21:163–169, August 1987.

[20] J. Manson, G. Petrova, and S. Schaefer. Streaming surface reconstruction using wavelets. *Computer Graphics Forum (Proceedings of the Symposium on Geometry Processing)*, 27(5):1411–1420, 2008.

[21] D. Reddy, A. K. Agrawal, and R. Chellappa. Enforcing integrability by error correction using l1-minimization. In *CVPR*, pages 2350–2357. IEEE, 2009.

[22] T. Simchony, R. Chellappa, and M. Shao. Direct analytical methods for solving poisson equations in computer vision problems. *IEEE TPAMI*, 12:435–446, May 1990.

[23] R. Zhang, P. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE TPAMI*, 21:690–706, August 1999.