Fast and High Quality Fusion of Depth Maps

Christopher Zach
Department of Computer Science
University of North Carolina at Chapel Hill

Abstract

Reconstructing the 3D surface from a set of provided range images – acquired by active or passive sensors – is an important step to generate faithful virtual models of real objects or environments. Since several approaches for high quality fusion of range images are already known, the runtime efficiency of the respective methods are of increased interest. In this paper we propose a highly efficient method for range image fusion resulting in very accurate 3D models. We employ a variational formulation for the surface reconstruction task. The global optimal solution can be found by gradient descent due to the convexity of the underlying energy functional. Further, the gradient descent procedure can be parallelized, and consequently accelerated by graphics processing units. The quality and runtime performance of the proposed method is demonstrated on well-known multi-view stereo benchmark datasets.

1. Introduction

The generation of high-quality, dense geometric models from passive or active sensors is still an active research topic. There is a large amount of research devoted to the task of surface reconstruction from a sparse or semi-dense set of given 3D points. In order to handle meshes with arbitrary genus, a volumetric representation is often utilized as the underlying data structure. Voronoi cells are the basis for some geometric methods for surface reconstruction [2, 1]. Regular volumetric grids are far more common for this purpose. Level set approaches [30, 29] fall into this category, although they are conceptually different than the approaches discussed next. To our knowledge none of these yet mentioned methods was employed on larger scale real-world data sets potentially containing a substantial amount of noise and outliers.

With the increased availabilty of laser scanning devices enabling dense measurements of real-world surfaces, several methods were developed to create full 3D models from such 2.5D range data. An explicit polygonal approach to merge several range images is described in [22].

Early range image fusion methods incorporating a regular voxel space include Curless and Levoy [8], Hilton et al. [10] and Wheeler et al. [24]. These methods initially compute a signed distance function for the final surface by averaging the distance fields induced by the given range images. The boundary representation of the resulting surface is obtained by an isosurface polygonization method. The merged distance field is computed separately on voxels, which allows those methods to be efficient, but spatial coherence and smoothness of the resulting mesh cannot be enforced. Volumetric graph cuts for computer vision [23] allow the incorporation of surface regularization into the volumetric fusion framework. Hornung and Kobbelt [12] present a general surface reconstruction method for sparse and dense input point clouds. Since their approach is based on a geodesic problem formulation, a constrained optimization scheme using conservative interior/exterior estimates is required to avoid degenerate solutions. Other recent work on volumetric surface reconstruction [13, 14] directly estimates the corresponding characteristic function from (oriented) point samples. Lempitsky and Boykov [15] present a global shape fitting method, that shares several elements with our approach (see the TV-Flux method described in Section 2), but involves a sophisticated graph cut procedure for discrete optimization.

Finally, several methods attempt to reconstruct an object surface directly from captured image data, thereby avoiding the use of active sensors. Such *multi-view stereo* approaches enable a convenient procedure for 3D reconstruction, and allow the creation of virtual representations on a larger scale. An important group of methods estimates the object surface directly from a consistency score for voxels (most notably, a photo-consistency measure [19, 26, 23, 21, 11, 3, 4]). Other approaches compute intermediate depth maps from small-baseline stereo in the first instance and use multiple depth images to perform a final surface integration step. Strecha et al. [20] and Merrell et al. [16] clean the initially obtained depth maps by fusing information from neighboring views. The latter work is targeted on real-time large scale reconstruction, hence runtime performance was considered more important than the final mesh

quality. Up to now [16] was the only reported approach to generate dense 3D geometry in less than one minute for benchmark datasets [18].

The method proposed in this work is an approach for general range image integration based on total variation shape denoising. It can be easily extended to a surface reconstruction procedure for sparse oriented point clouds. Since the main application of our work is object reconstruction from multiple views, we focus on captured images augmented with calibration and orientation information as the primary input data. Depth maps can be obtained by small-baseline stereo methods from image data. Depending on the image content and on the utilized dense stereo method, the resulting depth maps may contain a substantial amount of outliers and (usually non-Gaussian) noise. Thus, high-quality generation of full 3D models from depth maps is only possible with a robust approach.

This work extends the approach presented in [27] is several aspects: first, the distance based data fidelity term proposed in [27] is replaced by an histogram based one, allowing the whole procedure to be accelerated by modern graphics processing units. Further, a visual comparison of the proposed approach with a continuous formulation of the global shape fitting energy [15] is provided. Finally, results and runtimes for the large evaluation datasets [18] are presented.

2. Accelerated Range Image Fusion

The input of the main method is a set of potentially noisy range images. For best runtime performance, we employ a GPU-based implementation of a plane-sweep stereo method [25, 7, 28] to obtain these initial depth maps. Since this method is a purely local approach to dense stereo, one can expect a substantial amount of noise and mismatches in the resulting depth images. This set of depth maps provides 2.5D geometry for each view, which are subsequently converted to truncated signed distance fields (denoted by $f_i : \Omega \rightarrow \mathbb{R}$ for a voxel space Ω). We use a simple z-comparison to obtain a fast and suitable approximation of the true distances.

In [27] the implicit representation of the final surface is obtained as the spatially regularized median of the provided 3D distance transforms, i.e. a TV-L^1 energy consisting of a total variation part, $\int |\nabla u| d\vec{x} = \int \|\nabla u\|_2 d\vec{x}$, and an L^1 (i.e. absolute differences) term,

$$E^{TV-L^1}(u) = \int_\Omega \left\{ |\nabla u| + \lambda \sum_i |u - f_i| \right\} d\vec{x}, \quad (1)$$

is minimized with respect to u for the given set of distance transforms f_i. The resulting function $u : \Omega \rightarrow \mathbb{R}$ is the signed distance to the fused model, and the corresponding surface representation can be extracted by any isosurface polygonization method. The optimization procedure presented in [27] is efficient and globally optimal, but not particularly suited to be accelerated by a GPU. Specifically, the generalized thresholding procedure described in Proposition 2 in [27] requires a varying amount of computation for each voxel, which typically yields to a suboptimal performance on GPUs.

In this work we modify the TV-L^1 approach in two aspects: first, the data fidelity $\sum_i |u - f_i|$ is replaced by a more GPU-friendly approximation using weighted distances to evenly spaced representative values c_j; and second, the expensive generalized thresholding step performing an optimal line search is replaced by a simpler and faster descent step.

We assume that the provided distance fields f_i are bounded to the interval $[-1, 1]$. Hence we sample this interval by evenly spaced bin centers c_j, and approximate the original data fidelity term $\sum_i |u - f_i|$ by $\sum_j n_j |u - c_j|$, where and n_j are the respective frequencies. Thus, f_i is replaced by its closest bin center c_j in the data fidelity term. Note that the data term is integrated over the voxel space, hence a histogram is maintained for each voxel.

In contrast to the original TV-L^1 formulation those frequencies can be weighted arbitrarily. Particularly, this allows us to reduce the influence of depth values errorneously reported behind the true surface. Section 4 illustrates the impact of this modification. The overall energy to be minimized is then

$$E^{TV-Hist}(u) = \int_\Omega \left\{ |\nabla u| + \lambda \sum_j n_j |u - c_j| \right\} d\vec{x}. \quad (2)$$

Analogous to the procedure derived in [27], E^{Hist} is replaced by the strictly convex relaxation

$$E_\theta^{TV-Hist}(u, v) = \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - v)^2 + \lambda \sum_j n_j |v - c_j| \right\} d\vec{x}, \quad (3)$$

for a small value of θ. Since this energy functional is (strictly) convex in u and v, a global minimizer can be determined by alternating optimization with respect to u and v. Reducing the energy with respect to u and fixed v can be performed e.g. by Chambolles method [5]. We briefly review this method, which provides a global minimizer for the Rudin-Osher-Fatemi energy [17],

$$E_\theta^{ROF}(u; v) = \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - v)^2 \right\} d\vec{x}. \quad (4)$$

Note, that we omitted the histogram term, since it does not depend on u and therefore has a constant value. $|\nabla u|$ can

be rewritten as

$$|\nabla u| = \max_{\vec{p}:|\vec{p}|\leq 1} \langle \vec{p}, \nabla u \rangle. \quad (5)$$

Plugging Eq. 5 into the ROF energy (Eq. 4), and after computing the functional derivatives wrt. u and $\vec{p}$, we obtain the following conditions for stationary points:

$$\frac{\partial E_\theta^{ROF}}{\partial u} = \frac{1}{\theta}(u - v) - \nabla \cdot \vec{p} \stackrel{!}{=} 0, \text{ i.e.} \quad (6)$$

$$u = v + \theta(\nabla \cdot \vec{p}), \text{ and} \quad (7)$$

$$\frac{\partial E_\theta^{ROF}}{\partial \vec{p}} = \nabla u + \alpha \vec{p} \stackrel{!}{=} 0. \quad (8)$$

We employ a gradient descent/projection approach [6], hence $|\vec{p}| \leq 1$ is enforced after the gradient descent step and we can omit the Lagrange multiplier α.

The minimization of E_θ^{Hist} with respect to v is equivalent to (note that we can ignore the constant total variation term now)

$$\min_v \int_\Omega \left\{ \frac{1}{2\theta}(u - v)^2 + \lambda \sum_j n_j |v - c_j| \right\} d\vec{x}. \quad (9)$$

We can carry out this minimization point-wise, since this energy does not depend on any derivative of v. Without loss of generality (and omitting the dependence on the current voxel $\vec{x}$), if $u \in (c_k, c_{k+1})$ and the stationary point

$$v^* = u + \lambda\theta \left(\sum_{j>k} n_j - \sum_{j \leq k} n_j \right) \quad (10)$$

is in (c_k, c_{k+1}), then v^* is the minimum argument. If $v^* \leq c_k$, then $\hat{v} = c_k - \varepsilon$ lowers the energy in Eq. 9 for a sufficiently small ε (although it is not a global minimum in general). We exclude the bin centers c_j as feasible values for v in order to avoid handling of special cases in the GPU implementation. If $v^* \geq c_{k+1}$, we choose $\hat{v} = c_{k+1} + \varepsilon$. In those (very rare) situations, when c_j is exactly the optimum, $\hat{v}$ will oscillate with the values $c_j \pm \varepsilon$. Altogether, the generalized thresholding step to determine v for a given u to reduce the energy in Eq. 9 reads as

$$v = \max\{c_k - \varepsilon, \min\{c_{k+1} + \varepsilon, v^*\}\} \quad (11)$$

with v^* defined in Eq. 10 and $u \in (c_k, c_{k+1})$.

These alternating descent steps to update u and v, respectively, can be efficiently performed on modern GPUs. We employ 8 bins with bin centers $c_j = 2j/7 - 1$ for $j \in \{0, \ldots, 7\}$, and the frequencies n_j are determined by simple voting for the closest bin center. We will denote this procedure as the *TV-Hist* approach.

Another method that can be accelerated by GPUs is the global shape fitting approach of Lempitsky and Boykov [15]. Their energy minimization is based on discrete graph cuts, which is hard to parallelize. The utilized surface regularization energy is exactly equivalent to total variation regularization for binary functions. The data fidelity term is a flux energy derived from the provided input, a set of oriented, optionally sparse 3D points. Formulating these terms in a continuous setting, one arrives at the following resulting energy to be minimized:

$$E^{Flux}(u) = \int_\Omega \left\{ |\nabla u| + \lambda(\nabla \cdot \vec{f}) \cdot u \right\} d\vec{x}. \quad (12)$$

We just remark that (depending on the sign convention) the input scalar field $\nabla \cdot \vec{f}$ (divergence of the vector field $\vec{f}$) has positive values inside the presumed surface and negative values outside. We refer to [15] for more details. We will call this approach the *TV-Flux* method. As for the TV-Hist approach, a simple and efficient procedure to minimize the energy in Eq. 12 can be specified. It will be demonstrated in Section 4 that the resulting surfaces still contain substantial aliasing artefacts. This is due to the binary characteristics of the flux-based data term, since regions with non-vanishing flux strongly "push" to $+1$ or -1.

3. Implementation

In this section we briefly sketch our implementation of both approaches, TV-Hist and TV-Flux, on the GPU using OpenGL and Cg shader programs. Although the CUDA framework is a new and powerful GPU programming approach, traditional shader programming using OpenGL has still some benefits like more mature (and optimized) drivers and cached writes to render targets.

At first the input depth maps are converted to a volumetric data structure. For every voxel, an (approximate) signed distance to the surfaces induced by the depth maps is computed. This value is clamped to a user-supplied threshold and scaled to the interval $[-1, 1]$. We refer to [8, 27] for more details on depth map to volume conversion.

In the TV-Hist approach each voxel holds a histogram comprised of 10 bins. 8 bins are used to represent the distribution of values in the interior of the interval $(-1, 1)$, i.e. voxels close to the supposed surface. Two separate bins are reserved for values indicating a larger distance to the surface, one bin counts (scaled) distances smaller or equal -1 (occluded voxel) and one bin represents values greater or equal 1 (empty voxel). The total number of histogram counts is limited by the number of provided depth maps. In practice, the accumulated count is much smaller, since a specific voxel is typically only close to a depth surface for a subset of views. Thus, allocation of one byte for each histogram bin is sufficient.

We use the commonly employed flat layout of 3D volumes by mapping the z-slices to rectangular sections in a

2D texture. With a limit of 4096×4096 for the maximum texture size, voxel spaces slightly smaller than 256^3 can be handled (since we incorporate border pixels for the boundary conditions). Newer graphics hardware has a 8192×8192 limit enabling approximately 400^3 voxels. The memory requirements are far more constraining: for each voxel, the following data needs to be maintained:

1. 10 bytes for the input histogram,
2. 4 bytes for the current value of u (one float packed in four 8-bit values),
3. 6 bytes are required for the dual variable $\vec{p}$, since $\vec{p}$ is 3-dimensional. A 16-bit floating point representation for the elements of $\vec{p}$ is sufficient.

In total, the memory footprint of a voxel is 20 bytes (plus some additional buffers in the size of one slice). Thus, a 256^3 voxel space needs about 320Mb memory *on the GPU*. The memory requirement for the TV-Flux approach is 12 bytes per voxel.

Both methods, TV-Hist and TV-Flux, perform the following two alternating steps in the n-th iteration:

1. Update $\vec{p}$ using a gradient descent/projection approach (recall Eq. 8, $\partial E / \partial \vec{p} = \nabla u$), i.e.

$$\vec{q} \leftarrow \vec{p}_{(n)} + \frac{\tau}{\theta} \nabla u \tag{13}$$

$$\vec{p}_{(n+1)} \leftarrow \vec{q} / \max\left\{1, |\vec{q}|\right\}. \tag{14}$$

   $\tau < 1/6$ is the timestep of the gradient descent step.

2. Update u via intermediate computation of v. Here we provide a straightforward procedure to compute v^* (Eq. 10) suited for stream computing. For the TV-Hist approach, it consists of the following steps:

   (a) Compute a boolean (0/1) vector l by component-wise comparison,

$$l \leftarrow u_{(n)} > (c_j)_{j=0,1,\ldots,N},$$

   where c_j is the center of bin j ($c_0 = -\infty$). N is the total number of bins.

   (b) Compute a similar boolean vector

$$r \leftarrow u_{(n)} < (c_j)_{j=1,\ldots,N+1},$$

   with $c_{N+1} = +\infty$. The element-wise product $(l \cdot r)_j$ is then the indicator function $u_{(n)} \in (c_j, c_{j+1})$.

   (c) Compute $v^* \leftarrow u_{(n)} + \lambda\theta \left\langle r - l, (n_j)_j \right\rangle$, where $(n_j)_j$ is the histogram frequency vector.

   (d) Clamp v^*: Compute the interval borders

$$c_k \leftarrow \left\langle l \cdot r, (c_j)_{j=0,1,\ldots,N} \right\rangle$$

   and

$$c_{k+1} \leftarrow \left\langle l \cdot r, (c_j)_{j=1,\ldots,N+1} \right\rangle.$$

   Then $v \leftarrow \max\{c_k - \varepsilon, \min\{c_{k+1} + \varepsilon, v^*\}\}$.

   (e) Finally, $u_{(n+1)} \leftarrow v + \theta(\nabla \cdot \vec{p})$ (Eq. 7).

Note that this algorithm is already presented in a GPU-friendly manner, since it is based mainly on dot product and element-wise product calculations. The procedure to update u in the TV-Flux approach (Eq. 12) is simpler:

(a) Compute the intermediate value v from

$$v \leftarrow u_{(n)} - \lambda\theta(\nabla \cdot f).$$

(b) The new value for u is given by

$$u_{(n+1)} \leftarrow v + \theta(\nabla \cdot \vec{p}).$$

These two steps to update $\vec{p}$ and u, respectively, can be mapped directly to correponding fragment programs. Since read/write access to the same texture buffer has undefined behaviour, the update procedures iterate over the slices of the voxel space using an additional temporary buffer. Note that the gradient and divergence computations must be mutually dual, i.e. if forward differences are used to determine ∇u, then $\nabla \cdot \vec{p}$ needs to be computed from backward differences [5].

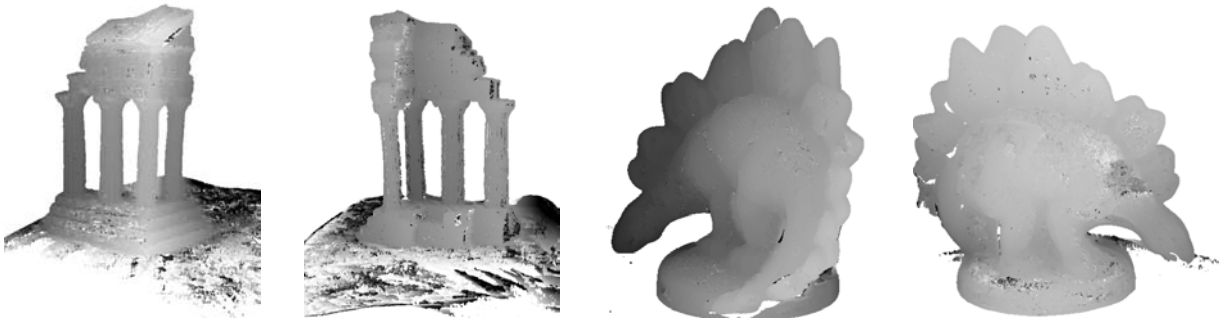
Both methods are still mostly bandwidth-limited, since gradient and divergence computations require sampling of several neighboring values. Hence the arithmetic intensity is not very high, and the simpler TV-Flux approach is in practice only slightly faster than the more complex TV-Hist method.

We want to point out that these schemata are embedded into a coarse-to-fine approach to obtain faster fill-in in regions with empty histograms (i.e. voxels distant to any surface induced by the depth maps).

4. Results

We show results on two datasets provided for benchmarking multi-view reconstruction methods [18] (see also `vision.middlebury.edu/mview/`). The advantage of demonstrating results on these datasets is, that the ground truth laser-scanned 3D geometry is known and the quality of obtained dense meshes can be evaluated.

We selected the full and the medium size data sets, called "Temple" (312 images), "Dino" (363 images), "Temple



(a) Temple ring depth #1 (b) Temple ring depth #2 (c) Dino ring depth #1 (d) Dino ring depth #2

Figure 1. Selected input depth maps used for later volumetric fusion. The depth maps are not very clean and contain a certain amount of mismatches.

ring" (47 images) and "Dino ring" (48 images). The respective images with a resolution of 640×480 are acquired in a controlled indoor environment, hence a simple and efficient SAD matching cost is sufficient to determine the input depth maps. 400 tentative depth planes are evaluated for dense stereo computation. The dark background pixels of the captured scene are removed by thresholding the intensity values. Pixels with a brightness value of at least 10 are considered to be foreground/object pixels, and background pixels otherwise. Figure 1(a)–(d) show a few samples of the depth maps obtained by a simple plane-sweep approach using a 3×3 SAD correlation window and two neighboring views as moving (warped) images.

Table 1 summarizes the characteristic numbers of these datasets, and depicts the runtimes observed to generate the intermediate depth maps and the fused 3D model, respectively. We further quote the runtimes for the actual (pure CPU-based) integration step from [27], where available. All timings are measured on PC hardware equipped with a NVidia GeForce 8800 Ultra GPU. Additionally, the accuracy results according to the main table of the Middlebury multi-view stereo page are displayed, too.

The surface meshes for both "ring" datasets are obtained with a value of λ equal to 0.08. The value of λ for the full datasets is scaled according to the number of images, i.e. $\lambda = 0.08 \frac{47}{312}$ for the full "Temple" dataset. θ is always set to 0.02, and in practice, the exact value of θ is not critical. The timestep τ is fixed to 0.16. We use three levels for the multi-scale pyramid, and 120 iterations are performed on every level. Although the energy has still not converged after this number of iterations, the extracted isosurfaces remain virtually constant. Figure 2(a)–(d) depict the obtained meshes for the "Temple" and "Temple ring" dataset.

The visual results for the "Dino" and "Dino ring" dataset, respectively, are illustrated in Figure 3. The overall geometry is captured very well. The "dent" visible in the back views (Figure 3(b) and (d)) is due to incorrect matches in the input depth maps (cf. Figure 1(d)). These mismatches appearing consistently in several depth maps are caused by a glossy specular reflection in an otherwise relatively homogeneous region. We can reduce the influence of those mismatches to some extent by reweighting the number of votes counting for empty voxels. In our experiments we used a weight of 0.25 for empty voxels. The surface step below the spikes is caused by an shadow edge visible in the source images. In our opinion, for the "Dino" datasets only the meshes generated by the time-consuming method proposed by Furukawa and Ponce [9] have a uniformly good appearance. It can be clearly seen from the resulting "Dino" meshes, that our simple choice of λ for the full dataset overestimates the visibility of surface elements, hence the respective model is slightly smoother than its "Dino ring" counterpart.

The quantitative evaluation comparing our results with the laser-scanned ground truth confirms the convincing visual impression: according to the main evaluation table (see [18] for the exact evaluation methodology), the accuracy values for the "Temple" and "Dino" meshes are 0.51mm and 0.55mm, respectively. The sparser ring datasets are accurate within 0.56 and 0.51mm. The completeness measures for all four meshes are consistently in the range of 98.7% to 99.1%. In terms of runtime, the only approach in the same performance category is [16], which is inferior in the obtained mesh quality. These numbers and the observed timing results place our proposed approach among the most efficient and high quality methods. Of course, these figures will vary if a different dense depth estimation method is employed.

Further, we provide a visual comparison between the TV-Hist approach and the TV-Flux method (Eq. 12), which is the continuous formulation of the global shape fitting energy [15]. Figures 4(a) and (b) depict the final mesh ob-

Dataset	Images	Voxels	Stereo	Integration	(CPU runtime [27])	Total	Accuracy	Completeness
Temple ring	47	$200\times300\times160$	8.8s	23.7s	(3m27s)	32.5s	0.56mm	99.0%
Temple	312		63s	83s	—	2m23s	0.51mm	98.8%
Dino ring	48	$200\times240\times200$	9.0s	26.4s	(3m50s)	35.4s	0.51mm	99.1%
Dino	363		70s	93s	—	2m43s	0.55mm	98.7%

Table 1. The illustrated datasets and their characteristic information.

(a) "Temple", front view (b) Back view (c) "Temple ring", front view (d) Back view

Figure 2. Final meshes for the "Temple" and "Temple ring" datasets.

tained for the "Temple ring" data, and Figures 4(c) and (d) show more detailed views of the models generated using TV-Flux and TV-Hist, respectively. The regularization weight is set to $\lambda = 0.3$ in order to obtain visually similar results. The overall geometry returned by the TV-Flux approach is comparable in quality with TV-Hist results, but closeup views reveal the inherent binary nature of TV-Flux solutions.

Finally, we provide results for outdoor image sequences. The first dataset consists of 16 views (see Figure 5(a)). The depth maps are computed by a local stereo approach using normalized cross correlation and a 5×5 support window (Figure 5(b)). Range image fusion is performed using a $320\times120\times200$ voxel grid. The obtained 3D model augmented with surface texture derived from the input images can be seen in Figure 5(c). A statue dataset with sample input images, intermediate depth maps and the final 3D model is displayed in Figure 6(a)–(e). The voxel grid resolution is $144\times288\times200$.

5. Conclusion and Future Work

In this work we present an efficient method for volumetric range image integration yielding high quality surface reconstructions from a given set of depth maps. The proposed procedure is very suitable to be performed on highly parallel working stream processors. In particular, we report runtime and accuracy evaluations for our GPU-accelerated implementation. We demonstrate that the achieved runtime performance is in the range of the currently fastest dense multi-view reconstruction method, while our approach returns very accurate 3D models at the same time.

Future work will address direct methods for multi-view stereo reconstructions. The photoflux shape optimization approach [3] is a promising starting point for such investigations. Again, a continuous formulation of the surface regularization energy will presumably lead to a simple and efficient method for multi-view reconstruction. Additionally, further research is needed to extend the class of smoothness terms that give rise to efficient and globally optimal algorithms.

References

[1] N. Amenta, S. Choi, and R. Kolluri. The power crust. In *Proceedings of 6th ACM Symposium on Solid Modeling*, pages 249–260, 2001.

[2] J.-D. Boissonnat and F. Cazals. Smooth surface reconstruction via natural neighbour interpolation of distance functions. In *Symposium on Computational Geometry*, pages 223–232, 2000.

[3] Y. Boykov and V. Lempitsky. From photohulls to photoflux optimization. In *British Machine Vision Conference (BMVC)*, pages 1149–1158, 2006.

(a) "Dino", front view (b) Back view (c) "Dino ring", front view (d) Back view

Figure 3. Final meshes for the "Dino" and "Dino ring" datasets.

[4] N. Campbell, G. Vogiatzis, C. Hernandez, and R. Cipolla. Automatic 3D object segmentation in multiple views using volumetric graph-cuts. In *British Machine Vision Conference*, 2007.

[5] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1–2):89–97, 2004.

[6] A. Chambolle. Total variation minimization and a class of binary MRF models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 136–152, 2006.

[7] N. Cornelis and L. Van Gool. Real-time connectivity constrained depth map computation using programmable graphics hardware. In *Proc. CVPR*, pages 1099–1104, 2005.

[8] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of SIGGRAPH '96*, pages 303–312, 1996.

[9] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *Proc. CVPR*, 2007.

[10] A. Hilton, A. J. Stoddart, J. Illingworth, and T. Windeatt. Reliable surface reconstruction from multiple range images. In *Proc. ECCV*, pages 117–126, 1996.

[11] A. Hornung and L. Kobbelt. Hierarchical volumetric multi-view stereo reconstruction of manifold surfaces based on dual graph embedding. In *Proc. CVPR*, pages 503–510, 2006.

[12] A. Hornung and L. Kobbelt. Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In *Eurographics Symposium on Geometry Processing*, pages 41–50, 2006.

[13] M. Kazhdan. Reconstruction of solid models from oriented point sets. In *Symposium on Geometry Processing*, pages 73–82, 2005.

[14] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Symposium on Geometry Processing*, pages 61–70, 2006.

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

[16] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nister, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *Proc. ICCV*, 2007.

[17] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[18] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR*, pages 519–526, 2006.

[19] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. *IJCV*, 35(2):151–173, 1999.

[20] C. Strecha, R. Fransens, and L. V. Gool. Combined depth and outlier estimation in multi-view stereo. In *Proc. CVPR*, pages 2394–2401, 2006.

[21] S. Tran and L. Davis. 3d surface reconstruction using graph cuts with surface constraints. In *Proc. ECCV*, pages 219–231, 2006.

[22] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of SIGGRAPH '94*, pages 311–318, 1994.

[23] G. Vogiatzis, P. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *Proc. CVPR*, pages 391–398, 2005.

[24] M. Wheeler, Y. Sato, and K. Ikeuchi. Consensus surfaces for modeling 3d objects from multiple range images. In *IEEE International Conference on Computer Vision (ICCV)*, pages 917 – 924, 1998.

[25] R. Yang and M. Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *Proc. CVPR*, pages 211–217, 2003.

[26] A. Yezzi and S. Soatto. Stereoscopic segmentation. *Int. Journal of Computer Vision*, 53(1):31–43, 2003.

[27] C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L^1 range image integration. In *Proc. ICCV*, 2007.

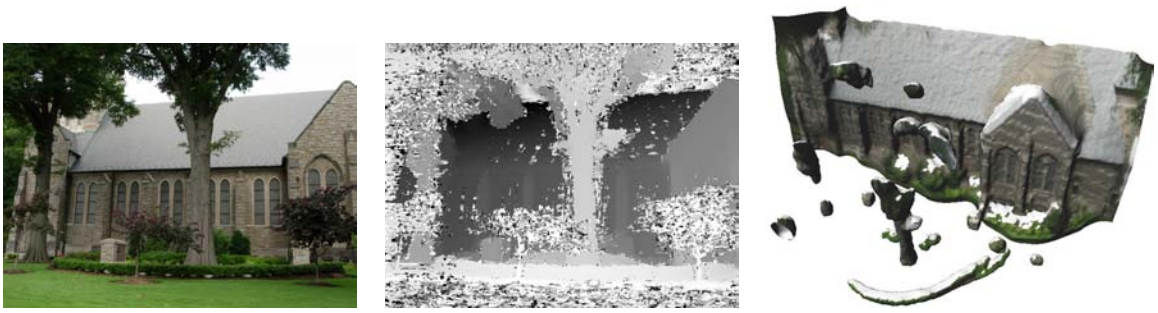
[28] C. Zach, M. Sormann, and K. Karner. High-performance multi-view reconstruction. In *Proc. 3DPVT*, 2006.

[29] H.-K. Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using the level set method. In *IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 194–201, 2001.

[30] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. *Computer Vision and Image Understanding*, 80(3):295–314, 2000.

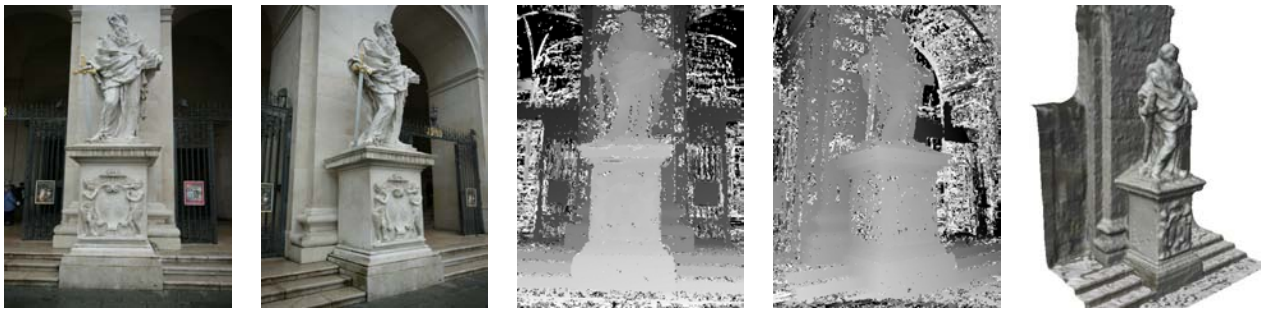
(a) TV-Flux, front view (b) Back view (c) TV-Flux, closeup view (d) TV-Hist, closeup view

Figure 4. Comparison of TV-Hist and TV-Flux energy formulations.



(a) One source image (of 16) (b) Sample depth map (c) Textured 3D model after fusion

Figure 5. An outdoor dataset consisting of 16 images. (a) depicts one source image. The depth maps (640×480) are noisy (b), and the finally obtained textured 3D model is displayed in (c).



(a) One source image (of 16) (b) One source image (of 16) (c) Sample depth map (d) Sample depth map (e) Textured 3D model after fusion

Figure 6. A statue dataset consisting of 16 images captured on a half-circle around the object. (a) and (b) are samples of the acquired images. Two depth maps are shown in (c) and (d). The obtained textured 3D model is displayed in (e).