Robust and Efficient Photo-Consistency Estimation for Volumetric 3D Reconstruction

Alexander Hornung and Leif Kobbelt

Computer Graphics Group, RWTH Aachen University
{hornung, kobbelt}@cs.rwth-aachen.de

Abstract. Estimating photo-consistency is one of the most important ingredients for any 3D stereo reconstruction technique that is based on a volumetric scene representation. This paper presents a new, illumination invariant photo-consistency measure for high quality, volumetric 3D reconstruction from calibrated images. In contrast to current standard methods such as normalized cross-correlation it supports unconstrained camera setups and non-planar surface approximations. We show how this measure can be embedded into a highly efficient, completely hardware accelerated volumetric reconstruction pipeline by exploiting current graphics processors. We provide examples of high quality reconstructions with computation times of only a few seconds to minutes, even for large numbers of cameras and high volumetric resolutions.

1 Introduction

Volumetric multi-view stereo reconstruction, originally introduced by Seitz et al. [1, 2], has recently been shown to produce 3D models from photographs or video sequences with fairly high quality [3, 4]. The basic principle in volumetric reconstruction is to find a classification for all elements (*voxels*) within a discretized volume whether they belong to the surface of the 3D object or not.

Probably the most central aspect of all these techniques is the estimation of the so called *photo-consistency* of a given voxel. The fundamental idea is that only voxels intersected by the object's surface have a consistent appearance in the input images, while other voxels project to incompatible image patches (Fig. 1). Currently there are two major approaches to this problem, either focusing on efficient computability or quality of the reconstruction.

Originally photo-consistency was measured based on the color variance of a voxel [1], assuming perfectly Lambertian and well textured surfaces under constant illumination conditions. Despite these restrictions this method is still widely used [5, 6] because of its computational efficiency and the often acceptable quality, e.g., for time-critical applications such as new view synthesis [7]. Since then the original approach has been improved in several ways. Bonet et al. [8] suggested extensions considering transparency. Zýka and Sára [9] present a statistical method for reliable outlier rejection. A probabilistic framework for space carving was presented by Broadhurst et al. [10]. Histogram-based color consistency tests were introduced by Stevens et al. [11], and Yang et al. [12] addressed the problems of textureless regions and specular highlights.

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part II, LNCS 3952, pp. 179–190, 2006.

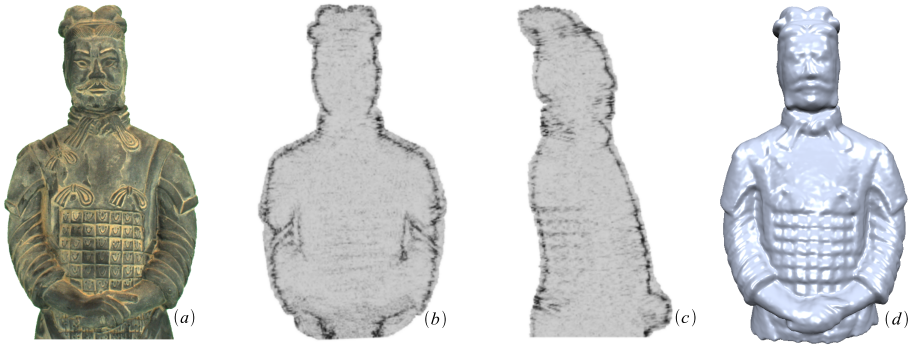

Fig. 1. An example for our improved photo-consistency measure for 3D reconstruction from images (a). Cuts through the computed consistency volume from a front and a side view for the warrior model are shown in (b) and (c) respectively. Darker colors indicate higher consistency values. The clear maximum at the actual surface location allows for reconstructed 3D models of high quality (d).

In recent work focusing on the quality of the reconstructed 3D model, photo-consistency is commonly evaluated based on more sophisticated consistency measures such as sum-of-squared-differences (SSD) or normalized cross-correlation (NCC) [13] of image patches instead of color variances. This greatly reduces ambiguous color configurations and accounts for changes in illumination because of the involved normalization step. Esteban et al. [3] present a technique based on deformable models, while Vogiatzis et al. [4] use global graph-cut optimization to find an optimal surface within a discretized volume that satisfies photo-consistency as well as smoothness constraints. Both methods achieve a very high quality of the reconstructed models. However both papers point out several open issues of NCC-based consistency estimation such as the question whether to use planar model- or image-aligned surface patches. In both cases projective warping can introduce a considerable matching error already for medium-baseline and non epipolar-aligned images. Our work resolves these restrictions based on a new, color normalized supersampling approach and specifically supports these recent optimization-based reconstruction techniques [3, 4].

A further important aspect besides the quality of a photo-consistency measure is its efficiency. Computation times up to several hours are common even in recent NCC-based work [3, 4] due to the much higher computational complexity. Although this could be considered acceptable with respect to the very high quality of the reconstructions, it is often still time-consuming to find optimal parameter settings in practice. Szeliski [14] addressed performance using adaptively refined grids. Partially hardware accelerated implementations of space carving were presented by Prock et al. [15] and Sainz et al. [16]. Solutions for hardware accelerated visual hulls and improved voxel visibility estimation have been discussed in [7, 17, 18]. Li et al. [7] presented a first completely hardware-based solution, Yang et al. [19] described a hardware-based SSD estimation for real-time stereo. However, these works either have conceptual

limitations in their applicability to recent optimization based approaches, or they have restrictions concerning the accuracy of the results or the complexity of the input data.

To resolve the above mentioned restrictions this paper presents a new implementation of the complete volumetric reconstruction pipeline. Most importantly, this includes a new approach to compute the photo-consistency of a voxel. Our consistency measure combines the advantages of the two above mentioned approaches, resulting in an illumination invariant, computationally efficient photo-consistency estimation for high quality 3D reconstruction. It improves robustness by resolving the problem of matching between surface samples even for completely unconstrained camera configurations, and is not restricted to planar surface approximations. We show how this consistency measure as well as all the other important stages of the volumetric reconstruction pipeline, namely visual hull and visibility determination, can be implemented in a highly efficient way by exploiting current graphics hardware, without any restrictions concerning the volumetric resolution, the number of images, nor the computational accuracy.

2 Photo-Consistency Estimation

Assuming fully calibrated, foreground segmented input images I_j of an object the general volumetric reconstruction pipeline consists of the following steps:

For each voxel v within a discretized volume one first has to determine whether it is contained in the visual hull of the object or if it lies in irrelevant parts of the volume. Voxels projecting to the background in one of the images I_j can be instantly marked as unoccupied space and skipped by further computations. We present an efficient background rejection test to estimate the object's visual hull in Sect. 3.1.

As emphasized by Vogiatzis et al. [4] the next important step is to use an initial geometry proxy such as the visual hull to determine whether a voxel v is visible in an input image I_j, or if it is occluded by other voxels. For basic visibility information one can compute approximate normals for each v by estimating tangent planes at the visual hull boundary and propagating the resulting normal directions inwards through the remaining volume. However, one additionally has to account for occlusions caused by other voxels. We present an efficient solution for this problem in Sect. 3.2.

After these initial steps we know in which images I_j a voxel v is visible. There exist two major approaches for the actual photo-consistency estimation which we will briefly introduce here to motivate our modified consistency measure.

Generally the photo-consistency $\phi(v)$ of a voxel is computed by comparing image patches P_j resulting from projecting v into images $I_j, j \in \{0, \ldots, N-1\}$ where v is visible according to the above mentioned visibility estimation. The original space carving approach [2] computes the color of a voxel v in image I_j as the average color c_j of all pixels $p_j^i \in P_j$, and computes $\phi(v)$ by applying a transfer function f to the color variance:

$$c_j = \frac{1}{|P_j|}\sum_i I_j(p_j^i), \qquad \overline{c} = \frac{1}{N}\sum_j c_j, \qquad \phi(v) = f\left(\frac{1}{N}\sum_j (c_j - \overline{c})^2\right). \quad (1)$$

This variance-based photo-consistency measure supports efficient computation and unconstrained camera setups. However, it is quite sensitive in practice to non-Lambertian, weakly textured surfaces, and varying illumination.

A more sophisticated approach used in recent work [3, 4] is to compare the intensity functions resulting from projecting v to images I_j and I_k by (normalized) cross-correlation (NCC). Suppose we approximate the unknown surface s intersecting voxel v by a planar surface patch (Fig. 2 a). The respective intensity functions can be compared by placing m object space samples p^0 to p^{m-1} on this patch, and evaluating their respective image space projections p_j^i and p_k^i, $0 \leq i < m$ in images I_j and I_k. Since s is unknown one generally computes an approximate solution by doing a pixel-wise comparison of simple, image-aligned patches P_j and P_k instead:

$$\mathbf{c}_j = (I_j(p_j^0) - c_j, \ldots, I_j(p_j^{m-1}) - c_j)^T, \quad \hat{\mathbf{c}}_j = \tfrac{\mathbf{c}_j}{\|\mathbf{c}_j\|}, \quad \phi(v) = f\left(\hat{\mathbf{c}}_j^T \cdot \hat{\mathbf{c}}_k\right), \quad (2)$$

with $p_j^i \in P_j$, $m = |P_j|$, c_j as defined in (1), and f being a transfer function applied to the NCC of P_j and P_k. This method strongly reduces potential color ambiguities and accounts for changes in illumination due to the involved normalization step. But despite these advantages there remains a number of open issues with this approach.

While the NCC is computed for pairs of image patches only, one has to combine results for more than two images to compute the actual photo-consistency ϕ. Vogiatzis et al. [4] propose to compute the average NCC for all image pairs, while Esteban et al. [3] compute the NCC with a single reference image. But more importantly one of the main problems of the above approach is the fact that pixels p_j^i and p_k^i in the images I_j and I_k respectively might not correspond to the same surface sample in object space. Hence image-aligned patches provide acceptable results only for medium baseline, epipolar-aligned images while setups with arbitrary camera configurations are difficult to handle. On the other hand as mentioned by Esteban et al. [3] more sophisticated planar model-aligned patches provide valid results only if the approximation is already quite close to the true object surface (Fig. 2 a).

2.1 Voxel Supersampling

To overcome the aforementioned problems we propose a new approach to create consistent *object* space samples p^i such that the matching error does not depend on the quality of the current surface approximation or view alignment but only on the volumetric resolution of the voxel grid.

Photo-consistency can be considered as a function $\phi(x, y, z)$ defined in continuous 3-space where the scene to be reconstructed is embedded. This function vanishes for points (x, y, z) lying exactly on the surface s, has small values in its immediate vicinity, and has larger positive values (which however do not

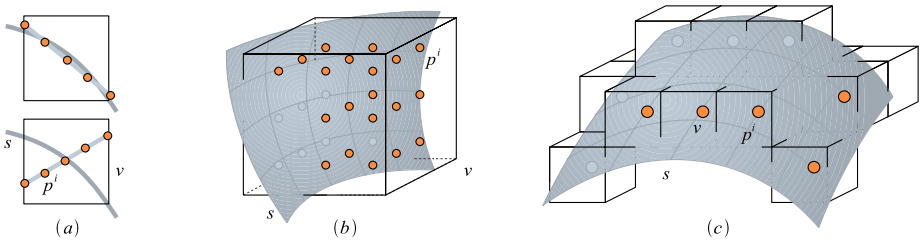

Fig. 2. For previous patch based methods the sampling error strongly depends on the approximation quality of the planar geometry proxy to the surface s (a). Our photo-consistency estimation is based on spatially supersampling a voxel v (b). The samples p^i are weighted equally since the exact position and orientation of s cannot be predicted at sub-voxel accuracy. At higher resolutions our approach allows us to use non-planar surface approximations at v (c) for the photo-consistency estimation.

necessarily increase for larger distances) everywhere else. If we do not have any reliable information about the exact location of the surface s within a given voxel, the best consistency indicator that we can check is to simply integrate the function $\phi(x, y, z)$ over the whole interior of the voxel. The value of this integral is expected to be relatively small in those voxels that are intersected by the surface. Obviously the integration of ϕ has to be done numerically, i.e., by supersampling the considered voxel at sub-voxel resolution (Fig. 2 b).

Within each voxel v we therefore uniformly distribute m equally weighted samples p^i in object-space and compute the colors for each of these samples separately by projecting them into the respective input images. This approach effectively eliminates the matching problem between the different images and samples even for completely unconstrained camera positions. To preserve the illumination invariance of the NCC-based approach we apply a similar color normalization step $\mathbf{c}_j^i \rightarrow \hat{\mathbf{c}}_j^i$ as in (2) to the colors of all 3D samples p^0 to p^{m-1} in a particular image I_j.

Instead of a pairwise correlation estimation which can either be biased by the reference camera [3] or which introduces an $O(n^2)$ complexity to evaluate all pairs [4] for each sample p^i, we compute a weighted variance of the normalized colors $\hat{\mathbf{c}}_j^i$ over all images. This allows us to take a weighted contribution of all images into account simultaneously, with the possibility to respect effects such as blurring at grazing viewing angles. We weigh the contribution of each image I_j to a voxel v using a Gaussian weight w_j (with $\sum_j w_j = 1$) of the angle between the approximate voxel normal and the voxel-to-camera direction in 3D space.

The final photo-consistency is simply computed as the sum of normalized color variances per sample:

$$\phi(v) = \frac{1}{m} \sum_i \phi(p^i), \quad \phi(p^i) = \mathrm{VAR}_j(w_j \hat{\mathbf{c}}_j^i) = \sum_j w_j (\hat{\mathbf{c}}_j^i)^2 - \left(\sum_j w_j \hat{\mathbf{c}}_j^i \right)^2 . \quad (3)$$

If we want to consider the full three channel color space instead of just one intensity channel, the number of input images n simply increases to $3n$.

2.2 Surface Sampling

The above supersampling approach provides a robust consistency measure as long as the projection of a voxel covers at least a few pixels in the input images. However, if the object space voxels are too small relative to the pixel resolution of the images this method tends to become unstable due to alias errors, e.g., when applying bilinear interpolation of color values (Fig. 5 c). Hence we have to enlarge the integration domain in this case by adding neighboring voxels. If we have additional information about which neighboring voxels are probably intersected by the surface, e.g., in an iterative optimization setting, we are in fact able to use non-planar geometry proxies for the consistency estimation.

Once an initial surface approximation is available it is straightforward to compute the k-nearest neighbor voxels which are intersected by the surface. E.g., for a technique such as [4] we can easily compute a signed distance field from the current surface within the remaining volume. Then the corresponding k-nearest neighbors for each voxel are found among its neighbors lying on the same level set.

Instead of supersampling a single voxel v we can now create samples p^i for each of the m closest neighbor voxels (Fig. 2 c) and simply compute the photo-consistency as described in Sect. 2.1. While this is conceptually similar to the patch-based NCC, we can exploit a non-planar surface approximation in contrast to planar patches using NCC. Again, the matching problem is implicitly avoided. This approach results in smooth surface reconstructions even at high volumetric resolutions relative to the resolution of the input images (Fig. 5).

3 Efficient GPU-Based Implementation

In comparison to the most simple form of NCC-based approaches our method introduces additional computational overhead since we have to compute the projections of each of the object space samples p^i instead of only the voxel center. In this section we will show how to compensate this overhead by exploiting the capabilities of programmable commodity graphics hardware.

The main benefit of using GPUs as general purpose processors is their inherent parallel processing capability. As we will show, our presented photo-consistency measure as well as further important steps during volumetric reconstruction can be effectively parallelized, resulting in significantly reduced processing times by using current GPU-features [20] such as vertex and fragment shader, floating point support, and efficient multi-resolution texture processing.

The underlying idea when transferring an arbitrary algorithm to the GPU is to exploit the possibility to execute a custom program for each generated vertex and fragment independently and in parallel instead of using the standard 3D rendering pipeline. Because of the floating point support of recent GPUs even quite complex input data can be processed by encoding it in the color channels of one or more textures. By simply drawing a screen-sized quad we generate $w \times h$ fragments on which a custom algorithm is executed. This means we effectively run this algorithm on the texture encoded input data $w \times h$-times in

one single rendering pass. The output data of the algorithm can then be accessed by reading it from the color channels of the framebuffer. The following sections present our implementation of a fully hardware accelerated reconstruction pipeline. Our OpenGL-based shader implementations are available on our webpage *http://www.rwth-graphics.de*.

In the following we assume that the volumetric scene representation is based on an adaptively refined grid (adaptive octree), and that we have pre-computed a multi-resolution pyramid of each input image [13]. Although the following algorithms explicitly address the multi-resolution capabilities of modern GPUs, they can be easily simplified to single resolution versions.

3.1 Visual Hull Estimation

For efficient voxel rejection based on segmented images we use a floating point texture T_p to encode the 3D position p for each voxel v in the (r,g,b)-channels of a single texture element (texel). Furthermore we initialize a texture T_b with a *false*-entry for each v as a boolean background mask. To avoid the complex estimation of a voxel's projected area P_j we load a texture mipmap T_I for each image I_j to the GPU and perform a single multi-resolution texture lookup in T_I such that $|P_j| \approx 1$. Projection matrices and the voxel size are transferred as environment parameters.

As described above we can execute a custom fragment program for each voxel v by drawing a screen-sized quad such that each v is represented by a single fragment f. The 3D position of each v is retrieved by a texture lookup $p := T_p(f)$. Then the projected position p_j and footprint size s_j of v in I_j are computed and a texture lookup $b_j := T_I(p_j, s_j)$ is used to check whether v is projected to the background in I_j. The results for all images are accumulated by updating the boolean background mask $T_b(f) := b_j \vee T_b(f)$ which is finally evaluated on the CPU. Since combined reading and writing to a texture is not supported on current GPUs the accumulation step is implemented using OpenGL framebuffer objects and two textures as alternating rendering targets [20].

The amount of voxels which can be encoded into a texture is limited by the maximum available texture size. Thus we run this algorithm repeatedly until all voxels are processed. For n images, v voxels, and a texture size of $w \times h$, this algorithm needs v/wh iterations with n image uploads for each pass.

3.2 Visibility Estimation

The next important step is the voxel visibility estimation based on the visual hull boundary V. The following approach is inspired by the ideas of GPU-based splat rendering by Botsch et al. [21] and uses techniques similar to their splat-based shadow-mapping, resulting in reduced processing times by several orders of magnitude in comparison to a standard approach such as ray-casting (Table 1).

The difficulty lies in choosing a proper occlusion surface for computing the visibility of voxels $v \in V$, since the thickness of V is more than one voxel. How-

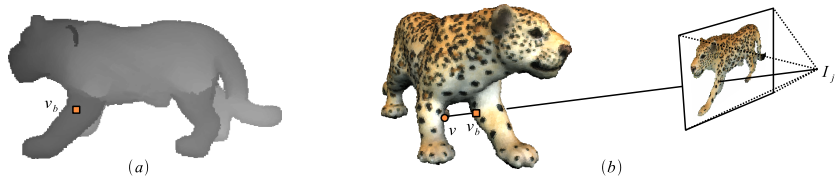

Fig. 3. For the visibility estimation of a voxel v in image I_j we first store the depth values of all backfacing voxels v_b in a depth map T_d (a). The visibility of each v can then be evaluated by a depth comparison of v and the corresponding entry in T_d (b).

ever this problem can be effectively solved using only the backfacing boundary of V. After computing the visual hull and normals as described in Sect. 2 we set the OpenGL projection matrix to the corresponding projection matrix of I_j and render all backfacing $v \in V$ as splats (circular discs in object space) into the depth buffer T_d. The splat radius is set in correspondence to the voxel size. The result is a dense depth map (Fig. 3 a) of all outer boundary voxels on the backside of surface s as seen from image I_j. Then the visibility for all v in I_j can be computed efficiently by a simple depth comparison.

Similar to Sect. 3.1 a fragment program loads for each fragment f the corresponding voxel position $p := T_p(f)$. The depth d of p in eye-space can then be compared to the depth value d_V of the front-most boundary voxel projecting to the same image position using a simple texture lookup in T_d (Fig. 3 b). Then v is visible iff $d < d_V$. The number of necessary iterations is identical to Sect. 3.1.

3.3 Photo-Consistency Estimation

The GPU-based photo-consistency estimation is slightly more involved than the previous steps because of the supersampling and color normalization. Assume we create m samples p^i per voxel v (Fig. 2). We encode the data of each sample in a separate texel such that a single voxel v is represented by a sequence of m texels (Fig. 4). In addition to the 3D positions p^i we also store the normal directions in another texture T_n to compute individual camera weights w_j. Auxiliary attributes such as the range of texture coordinates for each v are stored in T_a. The visibility computed in Sect. 3.2 is stored in an occlusion texture T_o. Finally, the image I_j, the corresponding projection matrix, and the voxel size are transferred to the GPU as a texture mipmap T_I and additional environment parameters. Similar to Sect. 3.1 color integration is avoided by a lookup in the corresponding mipmap level of T_I such that $|P_j| \approx 1$. The accumulated color values for solving (3) are stored in a texture T_{accum}. A fragment f is generated for every sample p^i of each voxel. Then, for all images I_j and fragments f, we run the following algorithm:

1. Projection pass:
   (a) Compute sample color $\mathbf{c}_j^i := T_I(p_j^i, s_j^i)$
   (b) Compute camera weight w_j based on T_n and T_o
   (c) Store color and weight $T_c(f) := (\mathbf{c}_j^i, w_j)$

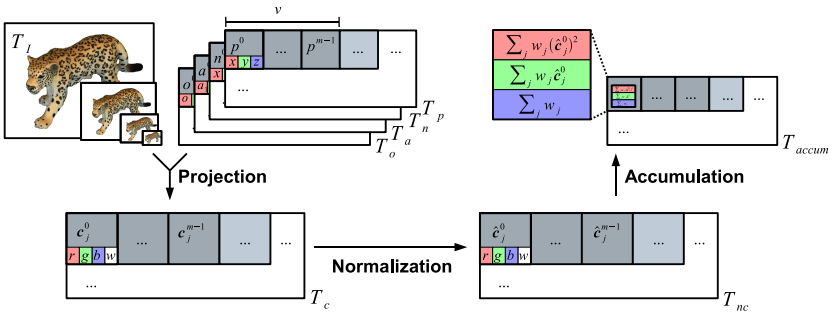

Fig. 4. Our hardware accelerated photo-consistency estimation is based on a three step rendering process. Using texture-encoded input data, we first compute projected color values for each sample p^i. These colors are then normalized and finally accumulated for the final consistency estimation.

2. Normalization pass:
   (a) Loop over all samples $\mathbf{c}_j^k, 0 \leq k < m$ (using T_a) and normalize $\mathbf{c}_j^i \rightarrow \hat{\mathbf{c}}_j^i$
   (b) Store normalized color and weight $T_{nc}(f) := (\hat{\mathbf{c}}_j^i, w_j)$
3. Accumulation pass:
   (a) Get $(\hat{\mathbf{c}}_j^i, w_j) := T_{nc}(f)$
   (b) Add $w_j \left(\hat{\mathbf{c}}_j^i\right)^2$, $w_j \hat{\mathbf{c}}_j^i$, and w_j to the accumulation buffer T_{accum}

Since we have three color channels per $\hat{\mathbf{c}}_j^i$ we accumulate the $3 + 3 + 1$ values computed in step 3b in two output buffers using multiple render targets [20]. The evaluation of these buffers and the summation over samples i in (3) is done in software since a GPU implementation would generate redundant summations for all fragments f corresponding to the samples of a single voxel. For n images, v voxels, m samples per voxel, and a texture size of $w \times h$, this algorithm needs vm/wh passes with n image uploads for each pass.

4 Results

The following section presents our evaluation of the presented method in terms of quality and efficiency. Our reference system for performance evaluation is a Linux-based Intel Pentium 4 with 3.2 GHz, 2 GB of main memory, and a NVIDIA GeFore 6800. We captured video sequences of the Warrior- (Fig. 1) and Leo-model (Fig. 5) with an uncalibrated turn-table setup and an image resolution of 1024×768. The Bahkauv-statue (Fig. 5) was captured using a hand-held video camera with an image resolution of 720×576. We pre-processed the video streams using standard structure-from-motion and segmentation techniques. All models were reconstructed by an iterative multi-resolution implementation of [4] consisting of our proposed volumetric reconstruction pipeline and a graph-cut based surface extraction at a volumetric resolution of 512^3.

The number of samples for each voxel was set to $m = 3^3$ for all experiments, approximately corresponding to a 5×5 image patch for NCC-based techniques.

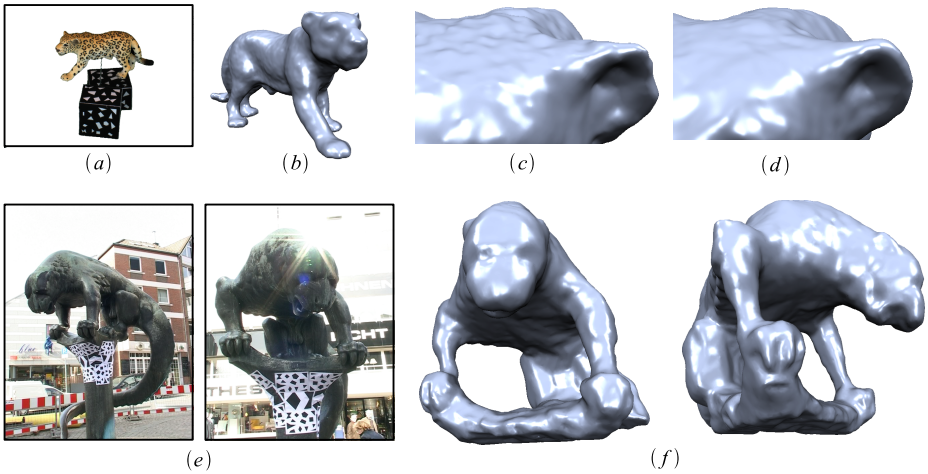

Fig. 5. Image (a) shows one of the original 46 input images of the Leo-model. The 3D model (b) was obtained using a graph-cut based technique [4]. Small oscillations and artifacts can occur for supersampled voxels projecting to less than a few pixels (c). Our surface sampling using neighboring voxels significantly improves the results (d). The approximate image size of the Leo-head is 140^2 pixels. The 30 images used for the reconstruction of the Bahkauv-statue (e) were captured using a hand-held video camera. We are able to reconstruct a quite detailed model (f) despite the specular surface and other illumination artifacts.

For lower values of about 2^3 samples particularly difficult areas such as the quite deep concavities of the Warrior's arms or small features such as the ears of the Leo model could not be properly reconstructed. For higher resolutions we did not observe a significant improvement of the reconstruction quality. However, our proposed surface sampling approach which includes neighboring voxels as discussed in Sect. 2.2 significantly improves reconstruction results for high voxel resolutions, so that one can achieve highly detailed, smooth reconstructions (Fig. 5) without the use of high resolution cameras. In our experiments we applied the surface sampling approach for volumetric resolutions, where a single voxel projects to less than 5^2 pixels. The reconstruction of the Bahkauv shows that a reconstruction is possible even under difficult lighting conditions with non-Lambertian, weakly textured surfaces.

Table 1 shows the performance of our GPU-based implementation in comparison to our CPU-based reference implementation. Although there is a certain overhead associated with loading images and voxel data to the GPU we achieve acceleration factors of 3 to 85. Using our multiresolution implementation of [4] the overall reconstruction time for all presented models was less than 10 minutes. Please note that computation times reported in related work [4, 3] range from about 40 minutes to several hours for comparable target resolutions and hardware. Our input data and results are available at *http://www.rwth-graphics.de.*

Table 1. Comparison of computed voxels per second for our hardware-based method and our software implementation (in parentheses) for different input complexities.

Images	Voxels v	Visual hull v/s	Visibility v/s	Consistency v/s	Total time
26	2M	19.4M (1.6M)	3.7M (55K)	350K (109K)	2.7m (24m)
26	4M	33.8M (1.7M)	4.6M (55K)	375K (122K)	5.1m (46m)
51	4M	42.0M (1.7M)	4.7M (56K)	423K (139K)	8.8m (87m)
126	4M	49.4M (1.8M)	4.7M (56K)	450K (139K)	20m (215m)
126	16M	50.2M (1.8M)	4.8M (56K)	450K (140K)	82m (858m)

5 Conclusion and Future Work

In this work we presented a new and efficient approach to compute the photo-consistency of voxels for volumetric 3D stereo reconstruction. Our method resolves several restrictions of previous methods such as the matching of surface patches, biased consistency estimation, and the necessity of epipolar-aligned images, while preserving important features such as illumination invariance. We showed furthermore how this consistency test as well as other important reconstruction steps can be efficiently implemented using commodity graphics hardware, leading to a fully hardware accelerated, high quality reconstruction pipeline for volumetric stereo.

As future work, we plan to incorporate methods to improve the handling of non-Lambertian surfaces. Although the Bahkauv-statue could be reconstructed with acceptable quality, we think that photo-consistency measures should explicitly model specularities and other surface properties [3, 12] for improved results.

Finally we could not yet exploit the full potential of our hardware implementation, since we observed a strong performance breakdown for texture sizes larger than 2048^2. This is probably related to the fact that some of the more recent OpenGL features still have open issues. Since the data transfer to and from the GPU is the main bottleneck of our method, we expect an approximately 4 times higher performance for texture sizes of 4096^2 because of the reduced number of iterations (and hence image uploads) for each algorithm.

Acknowledgements

We would like to acknowledge the helpful discussions with Mario Botsch, Martin Habbecke, and Volker Schönefeld.

References

1. Seitz, S.M., Dyer, C.R.: Photorealistic scene reconstruction by voxel coloring. In: CVPR. (1997) 1067–1073
2. Kutulakos, K.N., Seitz, S.M.: A theory of shape by space carving. International Journal of Computer Vision **38** (2000) 199–218

3. Esteban, C.H.: Stereo and Silhouette Fusion for 3D Object Modeling from Uncalibrated Images Under Circular Motion. PhD thesis, Ecole Nationale Supérieure des Télécommunications (2004)
4. Vogiatzis, G., Torr, P., Cipolla, R.: Multi-view stereo via volumetric graph-cuts. In: CVPR. (2005) 391–398
5. Sinha, S., Pollefeys, M.: Multi-view reconstruction using photo-consistency and exact silhouette constraints: A maximum-flow formulation. In: ICCV. (2005)
6. Slabaugh, G.G., Schafer, R.W., Hans, M.C.: Image-based photo hulls for fast and photo-realistic new view synthesis. Real-Time Imaging **9** (2003) 347–360
7. Li, M., Magnor, M., Seidel, H.P.: Hardware-accelerated rendering of photo hulls. Computer Graphics Forum **23** (2004) 635–642
8. Bonet, J.S.D., Viola, P.A.: Roxels: Responsibility weighted 3D volume reconstruction. In: ICCV. (1999) 418–425
9. Zýka, V., Sára, R.: Polynocular image set consistency for local model verification. In: Workshop of the Austrian Association for Pattern Recognition. (2000) 81–88
10. Broadhurst, A., Drummond, T., Cipolla, R.: A probabilistic framework for space carving. In: ICCV. (2001) 388–393
11. Stevens, M.R., Culbertson, W.B., Malzbender, T.: A histogram-based color consistency test for voxel coloring. In: ICPR. (2002) 118–121
12. Yang, R., Pollefeys, M., Welch, G.: Dealing with textureless regions and specular highlight: A progressive space carving scheme using a novel photo-consistency measure. In: ICCV. (2003) 576–584
13. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Prentice Hall (2002)
14. Szeliski, R.: Rapid octree construction from image sequences. Computer Vision, Graphics and Image Processing: Image Understanding **58** (1993) 23–32
15. Prock, A.C., Dyer, C.R.: Towards real-time voxel coloring. In: Image Understanding Workshop. (1998) 315–321
16. Sainz, M., Bagherzadeh, N., Susin, A.: Hardware accelerated voxel carving. In: SIACG. (2002) 289–297
17. Culbertson, W.B., Malzbender, T., Slabaugh, G.G.: Generalized voxel coloring. In: Workshop on Vision Algorithms. (1999) 100–115
18. Eisert, P., Steinbach, E., Girod, B.: Multi-hypothesis, volumetric reconstruction of 3-d objects from multiple calibrated camera views. In: ICASSP. (1999) 3509–3512
19. Yang, R., Pollefeys, M.: Multi-resolution real-time stereo on commodity graphics hardware. In: CVPR. (2003) 211–217
20. OpenGL extension registry. (http://www.opengl.org/)
21. Botsch, M., Hornung, A., Zwicker, M., Kobbelt, L.: High-quality surface splatting on today's GPUs. In: Eurographics Symp. on Point-Based Graphics. (2005) 17–24