# Dense 3D Reconstruction with a Hand-held Camera*

Benjamin Ummenhofer and Thomas Brox

Computer Vision Group
University of Freiburg, Germany
{ummenhof, brox}@informatik.uni-freiburg.de

**Abstract.** In this paper we present a method for dense 3D reconstruction from videos where object silhouettes are hard to retrieve. We introduce a close coupling between sparse bundle adjustment and dense multi-view reconstruction, which includes surface constraints by the sparse point cloud and an implicit loop closing via the dense surface. The surface is computed in a volumetric framework and guarantees a dense surface without holes. We demonstrate the flexibility of the approach on indoor and outdoor scenes recorded with a commodity hand-held camera.

## 1 Introduction

Taking a video camera and walking around an object to automatically build a 3D object model has been a long-standing dream in computer vision. What seemed unreachable 20 years ago is now close to our abilities thanks to the many discoveries in 3D geometry and image matching over the years.

What currently seems to hinder the use of more 3D information in other domains is the specialization of 3D reconstruction methods to specific settings. State of the art multi-view stereo methods are able to produce accurate and dense 3D reconstructions. Unfortunately, many of them are tuned to settings like the Middlebury benchmark [15], where the camera poses are known and silhouette information can easily be retrieved. One of these methods is the approach of Kolev et al. [11], which we use as a basis for our approach. On the other hand, structure from motion approaches allow to estimate camera poses and sparse point clouds even from unordered photo collections [16, 4].

In this paper we aim to work towards closing the gap between sparse bundle adjustment and dense volumetric reconstruction by proposing a close coupling between the two. We build an initial sparse reconstruction of the scene with incremental bundle adjustment and point correspondences computed with the point tracker by Sundaram et al. [18]. The resulting point cloud is then integrated into an energy functional for dense reconstruction. The point cloud constraints are sufficient to define the coarse object volume. As a consequence, we do not require any silhouette information, which is hard to get in general scenes.

* We gratefully acknowledge partial funding by the ERC Starting Grant VIDEOLEARN.

Vice versa, the dense surface is used to improve the camera poses following an idea from Aubry et al. [1]. In our case, this also leads to a refinement of the point cloud constraints. In contrast to approaches based on depth map fusion, we obtain a fully consistent 3D object reconstruction without holes in the surface. In particular, the coupling between dense and sparse reconstruction leads to an implicit loop closing that corrects the initial errors of bundle adjustment due to drift in the point trajectories.

We show results on an indoor as well as two outdoor videos taken with a consumer camera demonstrating the flexibility of the approach.

## 2 Related Work

Apart from the above mentioned works and the references within the Middlebury benchmark [15] there are a couple of works, which are closely related to the approach we present in this paper.

Hiep et al. [9] also aims for dense reconstructions and efficiently handles large scenes. The first step in their pipeline is the creation of a point cloud with millions of points. The point cloud is then converted in a visibility consistent triangle mesh. As a last step, a variational method refines the photoconsistency of the mesh. In contrast to our approach, none of these steps uses feedback from the dense reconstruction to improve the initial motion or structure.

A method proposed by Goesele et al. [6] computes depth maps from internet photo collections. As initialization they use the structure from motion approach from [16]. The final dense reconstruction step is based on merging the computed depth maps.

Yezzi and Soatto [21] propose a method where the cameras are refined during the dense reconstruction process. However, their silhouette based approach is not well suited for general image sequences. It works best for textureless scenes where the background can be clearly separated from the object.

Furukawa and Ponce [5] describe an iterative camera calibration and reconstruction approach. A large set of patches is used to refine the camera poses. Vice versa the refined camera poses are used to improve the reconstruction. To compute the dense surface they use the method from Kazhdan et al. [10]. The resulting surface is fitted to the patches by the solution of a Poisson problem. As the spatial Poisson problem does not take photoconsistency into account, results suffer in areas not covered by sufficiently many patches.

Lempitsky and Boykov [12] present a shape fitting approach that maximizes the flux of the surface and a vector field induced by weakly oriented points in the min-cut framework.

Newcombe et al. [14] presented an interactive approach that allows for acquisition of dense 3D models in real-time. While [14] is primarily a camera tracker, the underlying dense depth maps can be used also for scene reconstruction. To track the camera, the image is aligned to a 2.5D scene representation based on depth maps. Multiple depth maps are combined to cover the scene. The approach shares the problems of other methods based on depth map fusion. Additionally,

since there is no global optimization of the surface, a consistent object reconstruction from all sides currently seems to be out of reach. A similar method has been represented in Graber et al. [7]. Instead of modelling the scene with a collection of depth maps, they fuse generated depth maps into a volumetric grid.

## 3 Sparse Initialization

This section describes the computation of initial camera poses $\mathcal{C}$ and a sparse point cloud $\mathcal{P}$ roughly indicating the structure of the scene. As input we assume an image sequence with known camera intrinsics. This is justified if we assume that the camera intrinsics do not change within the sequence, which is true as long as we do not use the zoom function.

To compute the structure of the scene we generate point correspondences using the tracker from Sundaram et al. [18]. The tracker generates point trajectories based on optical flow. Trajectories are generated frame by frame and errors in the optical flow accumulate, leading to significant drift in long trajectories. We will correct the errors due to this drift in Section 4. Short trajectories suffer less from drift but generate only small baseline measurements, which is disadvantageous for bundle adjustment. Therefore, we consider only trajectories with a minimum length of 50 frames

To reconstruct the whole scene we use a hierarchical approach. We divide the sequence into parts with up to 150 frames and perform incremental bundle adjustment on each of the parts. The parts are then recursively merged to obtain the final reconstruction. For bundle adjustment we use the implementation of Wu et al. [20]. After adding a new camera, we generate new points from the trajectories. Among all points we remove those with a reprojection error exceeding 10 pixels on images with a resolution of $1280 \times 720$. Fig. 1 shows the sparse reconstruction of a scene with multiple objects.

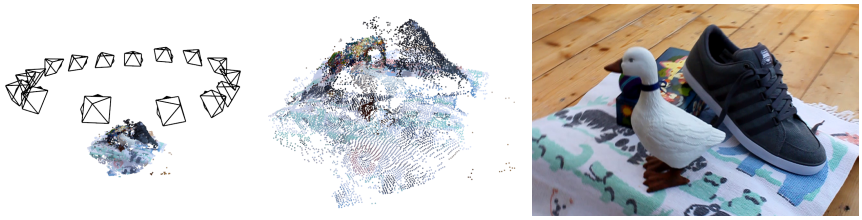

Fig. 1: **Left,Center:** Sparse reconstruction of the scene. The scene comprises multiple objects with complex visibility. The reconstruction contains 16125 points and 600 cameras. **Right:** One of the input images used for the sparse reconstruction

## 4 Dense Reconstruction

As in [11] we pose the reconstruction problem as a segmentation of a volume into object and empty space. Our energy functional comprises two zero-order terms and one first-order term. The zero-order terms $r_1$ and $r_2$ correspond to unary costs in a discrete setting while the first-order term $\rho$ acts as pairwise costs:

$$E(u) = \int \left(\alpha r_1(\mathbf{X}) + r_2(\mathbf{X})\right) u(\mathbf{X}) + \beta\rho(\mathbf{X})\|\nabla u\| \mathrm{d}\mathbf{X} \; . \tag{1}$$

The sought binary function $u \in \{0, 1\}$ assigns points to either empty space or object. The function $\rho(\mathbf{X})$ is a photoconsistency measure that indicates the cost when placing the surface at position $\mathbf{X}$. The term involving $\rho$ can be regarded as a weighted Total Variation (TV) norm. Since $\rho$ is positive everywhere, it favors minimal surfaces. The global optimum of the last term alone would be the trivial solution. This is why it must be accompanied by zero-order terms. $r_1$ imposes constraints based on the sparse reconstruction and is absolutely necessary to kick-start the reconstruction, whereas $r_2$ is a zero-order representation of the photoconsistency and helps in areas that are not sufficiently covered by initial 3D points. The inclusion of $r_2$ allows to set $\alpha = 0$ in the last iteration (Sec.4.4) and it yields better results as shown in Fig. 4.

In the two following sections we describe the computation of the functions $r_1, r_2$ and $\rho$, which we need to solve (1). In Sec. 4.3 we explain how to update the sparse reconstruction using the dense surface. Our minimization strategy that iteratively refines the sparse and dense reconstruction is described in Sec. 4.4.

### 4.1 Constraints by the Sparse Point Cloud

In the beginning of the reconstruction process the surface estimation fully relies on the sparse reconstruction. This means that the point set $\mathcal{P}$ together with the last term in (1) with $\rho = 1$ must drive the initial surface. We express the constraints that the surface should be close to the points $\mathcal{P}$ as a voxel-wise cost that can be integrated elegantly in (1).

A point together with a viewing direction clearly indicates the object's interior and exterior in the direct vicinity of the point: the volume in direction of the camera belongs to the exterior while the volume in the opposite direction must belong to the object. We mollify this constraint with anisotropic diffusion [19] where a diffusion tensor emphasizes diffusion in the viewing direction of the camera. Fig. 2 shows the costs generated from a single point-camera pair.

To compute the cost function $r_1$, all points and all cameras that observe a point are considered and superposed. We use the distinctive region costs shown in Fig. 2 as a lookup-table and superpose all point-camera pairs. This way the cost can be computed voxel-wise and is suitable for implementation on the GPU.

### 4.2 Photoconsistency Constraints

The photoconsistency function $\rho(\mathbf{X})$ measures the cost of placing the surface at $\mathbf{X}$. If we take a patch in one image and project it to the correct surface, the

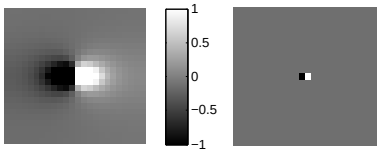


Fig. 2: **Left:** Central slice of the region cost volume generated by a point-camera pair. The size of the depicted volume is $24 \times 23 \times 23$. **Right:** The seed points with fixed values -1 and 1. The observed point is located between the seed points. The viewing direction of the observing camera is the negative x-axis

reprojections of the surface to any camera in which the surface is visible should be identical to the image observed in this camera apart from shading effects. In contrast, the same projection with a badly placed surface will be different to the observed image.

We compute the photoconsistency in the image space for each camera. The photoconsistency $P$ at a pixel $\mathbf{x}$ and depth $d$ is computed with normalized cross correlation in a $7 \times 7$ window

$$P_i(\mathbf{x}, d) = \frac{1}{|\mathcal{C}'| - 1} \sum_{j \in \mathcal{C}' \setminus i} \mathrm{NCC}(I_i(\mathbf{x}), R_i^j(\mathbf{x}, d)) \ , \tag{2}$$

where $I_i$ is the image of the $i$-th camera. The corresponding image $R_i^j$ is a rendering of a surface at depth $d$ with the projective texture of camera $j$. We use normalized cross correlation to be robust to most shading effects.

Computing the photoconsistency is the most expensive part of the dense reconstruction despite the use of the GPU's rendering functions to project each image to the surface and back to each camera. Moreover, since we work with a video, neighboring images are very close together and show very similar content. Hence, we select only a subset $\mathcal{C}' \subset \mathcal{C}$ of about 50 cameras. Additionally, we compute the photoconsistency only near the current surface estimate. In particular, we compute the signed distance function of the surface and generate triangle meshes for the level sets $\{-8, -7, \ldots, 8\}$. We render each of these 17 meshes $|\mathcal{C}'|(|\mathcal{C}'| - 1)$ times. We use shadow mapping to mask out those parts of the images that are not visible in the camera providing the texture.

Like in Hernández et al. [8] and Kolev et al. [11], each pixel in each camera casts a vote for the depth with maximal photoconsistency. All votes are accumulated to yield the cost function

$$\rho(\mathbf{X}) = \exp\left( -\lambda \sum_{i \in \mathcal{C}'} \delta(d_i^{\max} = \mathrm{depth}^i(\mathbf{X})) P_i(\pi_i(\mathbf{X}), d_i^{\max}) \right) \ , \tag{3}$$

where $\pi_i$ projects to camera $i$, $d_i^{\max}$ is the depth with maximum photoconsistency at $\pi_i(\mathbf{X})$ , and the Kronecker $\delta$ indicates whether the depth corresponds to the considered voxel $\mathbf{X}$. $\lambda = 0.1$ is a scaling parameter.

To provide a zero-order cost $r_2$ based on photoconsistency, we search for each voxel parallel to the gradient of the signed distance function for the point with minimum $\rho$. Let $\mathbf{X}_{\min}$ be the position where $\rho$ is minimal and $\Phi$ be the signed distance function, then the term $r_2$ is defined as

$$r_2(\mathbf{X}) = \begin{cases} \operatorname{erf}\left(\frac{\Phi(\mathbf{X})-\Phi(\mathbf{X}_{\min})}{\sigma\rho(\mathbf{X}_{\min})}\right), & -8 < \Phi(\mathbf{X}) < 8 \\ 0, & \text{else} \end{cases}, \tag{4}$$

where $\sigma$ is another scaling parameter. The photoconsistency cost $\rho(\mathbf{X}_{\min})$ steers the slope near the zero-crossing. A high cost indicates a high uncertainty in the position of the surface and $r_2$ will become flat near the zero-crossing to account for this. Note that we set $r_2$ to zero in those areas where we did not compute the photoconsistency. This way we make sure not to bias the reconstruction towards the current surface estimate.

### 4.3 Camera and Points Refinement

The reconstruction quality depends much on the accuracy of the estimated camera poses. The initial camera poses from the sparse reconstruction are subject to the accumulated error in the point trajectories. Fig. 3 shows the influence of the camera poses on the reconstruction quality.

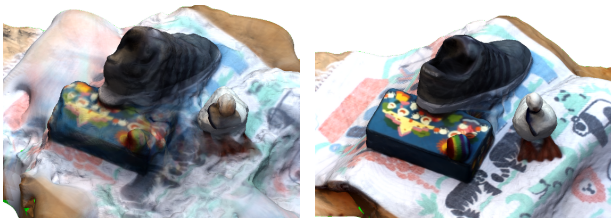

Fig. 3: **Left:** Dense, textured reconstruction based on the initial camera poses. The objects appear to be molten and the texture is blurred. **Right:** Dense reconstruction after refinement of the cameras

In classical bundle adjustment, the camera parameters are computed by minimizing the reprojection error of a sparse set of points. Now that we have a dense surface estimate, we can use all points on this surface to minimize a dense reprojection error. Recently, Aubry et al. [1] proposed to estimate this error by means of the optical flow between the textured rendering of the surface and the actual image. We stick to this idea using a GPU implementation of the optical flow from Brox et al. [3]. To generate the texture we simply average the projections from all cameras.

Given the optical flow, we minimize the reprojection error by formulating the problem as a pose estimation problem from 2D-3D point correspondences. The 3D points are generated from the surface estimate such that the projections of

the points are evenly distributed in the image. The corresponding 2D points are computed by projecting the 3D points to the image plane and adding the optical flow vector at that position as a correction. The camera pose, minimizing the reprojection error in a least squares sense, is computed using the algorithm by Lu et al. [13].

To account for the uncertainties in the optical flow estimation we verify the consistency between the forward flow $\mathbf{u}_{\text{fwd}}$ and the backward flow $\mathbf{u}_{\text{bwd}}$, and give 2D-3D point correspondences at inconsistent points less weight. The weight $w_i$ for a 3D point $\mathbf{X}_i$ and the corresponding 2D point $\mathbf{x}_i = \pi(\mathbf{X}_i) + \mathbf{u}_{\text{fwd}}(\pi(\mathbf{X}_i))$ is defined as

$$w_i = \frac{1}{1+b} \; ; \quad b = \frac{\|\pi(\mathbf{X}_i) - \mathbf{u}_{\text{bwd}}(\mathbf{x}_i)\|}{\|\mathbf{u}_{\text{fwd}}(\pi(\mathbf{X}_i))\| + \epsilon} \; , \tag{5}$$

where $\epsilon$ is a small constant and $b$ is the endpoint error relative to the length of the forward flow vector.

The refined camera poses allow us to update the point cloud $\mathcal{P}$. In contrast to the initial sparse bundle adjustment in Section 3, all cameras are mutually connected via the dense surface. Hence, we can use shorter trajectories, which are less affected by drift, to triangulate the 3D points.

### 4.4 Minimization

To minimize the energy in (1), we relax the binary function $u$ to take values in the range $[0, 1]$. This makes it a convex problem for given camera parameters and photoconsistency $\rho$. Since the computation of $\rho$ and the refinement of the camera parameters depends non-linearly on $u$, the overall problem is non-convex. Consequently, it can be optimized only locally by iterating the optimization with respect to $u$, the camera parameters, and $\rho$.

To minimize with respect to $u$ we decouple the zero-order terms from the first-order term

$$E(u, v) = \int \left(\alpha r_1(\mathbf{X}) + r_2(\mathbf{X})\right) v(\mathbf{X}) + \frac{1}{2\theta}\left(u - v\right)^2 + \beta\rho(\mathbf{X})\|\nabla u\| \mathrm{d}\mathbf{X} \; . \tag{6}$$

This leads to a simple point-wise optimization problem in $v$ and a standard weighted TV regularization problem in $u$. The latter can be solved efficiently with the numerical scheme from Bresson et al. [2]. The functions $u$ and $v$ are coupled by the term $\frac{1}{2\theta}(u-v)^2$. This decoupling approach allows for an efficient GPU implementation and has been used in many related problems [22, 17, 14].

To deal with the non-convexity of the overall problem, we employ a coarse-to-fine approach. For image sequences with a resolution of $1280 \times 720$ we suggest three levels with voxel grid resolutions of about $64^3$, $192^3$ and $320^3$. Similarly, we downsample the input images such that the height of the image is twice the largest extent of the grid.

At each level we repeatedly minimize (6) for fixed functions $r_1$, $r_2$, and $\rho$. After each of these inner iterations we refine the cameras $\mathcal{C}'$, the point cloud $\mathcal{P}$, and recompute the terms $r_1, r_2, \rho$ as described in the previous sections. We also

linearly decrease the parameter $\alpha$ from $\alpha = 5$ in the first iteration to $\alpha = 0$ in the last iteration. The idea is to give the sparse reconstruction a high weight in the beginning while we solely rely on the photoconsistency in the last iteration. This allows to reintroduce reconstruction details by $r_1$ that may not have been captured at coarser levels and to avoid reconstruction errors caused by errors in the points $\mathcal{P}$. Fig. 4 shows an example how the coarse-to-fine scheme may lose details when we set $\alpha = 0$ for all iterations on the two finest levels.

Fig. 4: **Left:** Reconstruction using $r_1$ and $r_2$. **Center:** Reconstruction without $r_1$: the head of the goose is too small to be represented at the coarsest voxel grid and cannot be recovered at finer levels due to local minima in $r_2$. **Right:** Reconstruction without $r_2$: the rear part of the shoe has been separated because the point density there is too low

## 5 Results

We recorded three sequences in different environments to demonstrate the robustness and usability of our approach. The sequences contain several images affected by motion blur and camera shake. Similar sequences could be captured by any non-expert with their personal camcorder.

Fig. 5 shows the recorded scenes and their reconstructions. The *shoe* sequence is an indoor scene with complex visibility as objects mutually occlude each other and some parts are only seen in few frames. The scene also demonstrates the effect of the sparse reconstruction that preserves details like the head of the goose. The *bird house* sequence highlights the importance of camera refinement and the implicit loop closing. While the camera motion for the *shoe* sequence is a complete ring around the object (see Fig. 1), the motion in *bird house* does not describe a closed ring. As a result, the reconstruction on one side is less accurate and the texture cannot be retrieved for the whole object. Another outdoor sequence *head* shows that we can also reconstruct objects that are not perfectly static.

Problematic for our algorithm are small delicate structures. This can be seen in the *shoe* sequence where the beak of the goose is missing or in the *head* sequence where the glasses are missing. In both cases the volume of the missing objects is small and the size of the objects in the images is small which is disadvantegous when computing the photoconsistency.

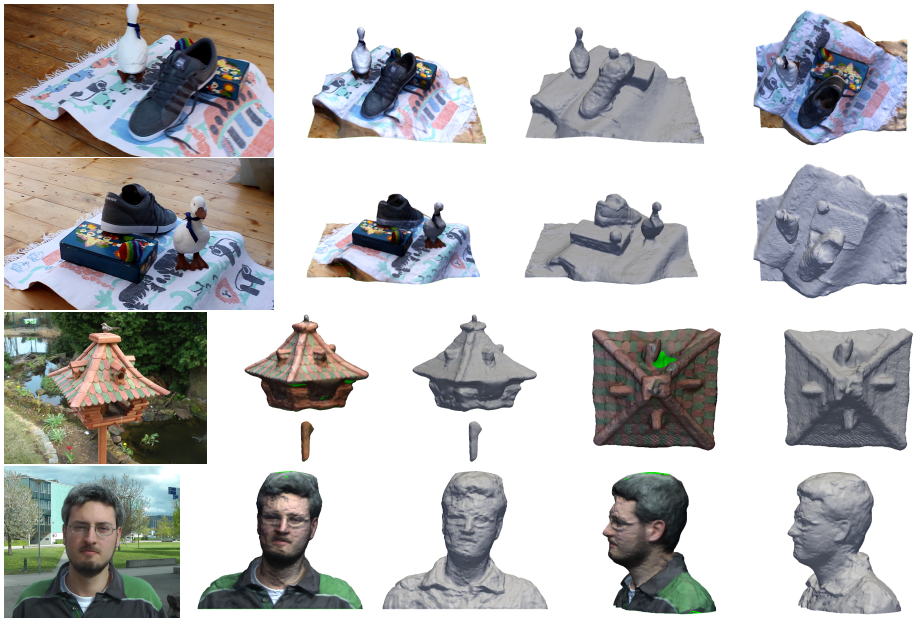

Fig. 5: **First, Second Row:** Indoor scene *shoe* with many occlusions. Even details such as the shoelaces are reconstructed. Grid size: [320 × 199 × 249]. **Third Row:** Outdoor scene *bird house*. Green spots on one side of the object in the textured views indicate missing texture, these parts are not seen in any image. The reconstruction of this side is less accurate than the others, but still a consistent reconstruction has been obtained. Grid size: [320 × 291 × 300]. **Fourth Row:** Outdoor scene *head*. Grid size: [320 × 247 × 266]

Table 1 lists the runtimes for the scenes in Fig. 5. All major parts of our algorithm use the GPU. The most time-consuming parts in the dense reconstruction are the photoconsistency computation (ca. 38%), the optical flow (ca. 22%) and minimization of (6) (ca. 10%).

Table 1: Runtimes on an Intel Xeon X5675@3GHz + GTX580

| | Frames | Tracking | Sparse | Dense $\left(\frac{\text{Iterations}}{\text{per Level}}\right)$ | Total |
|---|---|---|---|---|---|
| Shoe | 600 | 1h 32m | 3m | 1h 52m (5) | 3h 27m |
| Bird house | 270 | 0h 56m | 2m | 1h 59m (5) | 2h 57m |
| Head | 251 | 0h 36m | 1m | 1h 20m (5) | 1h 57m |

## 6 Conclusions

We have presented an approach for dense 3D reconstruction from image sequences in the absence of a controlled environment. The approach integrates sparse bundle adjustment into a dense variational formulation to provide an initialization that does not require silhouettes and ensures that important details

are not smoothed away. Thanks to the close coupling, both the sparse and the dense parameters benefit from mutual refinement.

## References


1. Aubry, M., Kolev, K., Goldluecke, B., Cremers, D.: Decoupling photometry and geometry in dense variational camera calibration. In: Proc. ICCV (2011)
2. Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J.P., Osher, S.: Fast global minimization of the active Contour/Snake model. Journal of Mathematical Imaging and Vision 28(2), 151–167 (2007)
3. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Proc. ECCV. LNCS, Springer (2004)
4. Frahm, J.M., Fite-Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.H., Dunn, E., Clipp, B., Lazebnik, S., Pollefeys, M.: Building rome on a cloudless day computer vision. In: Proc. ECCV. LNCS, Springer (2010)
5. Furukawa, Y., Ponce, J.: Accurate camera calibration from multi-view stereo and bundle adjustment. In: Proc. CVPR (2008)
6. Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.M.: Multi-View stereo for community photo collections. In: Proc. ICCV. pp. 1–8 (2007)
7. Graber, G., Pock, T., Bischof, H.: Online 3d reconstruction using convex optimization. In: Computer Vision Workshops (ICCV Workshops). IEEE (2011)
8. Hernández, C., Schmitt, F.: Silhouette and stereo fusion for 3d object modeling. Computer Vision and Image Understanding 96(3), 367–392 (2004)
9. Hiep, V.H., Keriven, R., Labatut, P., Pons, J.P.: Towards high-resolution large-scale multi-view stereo. In: Proc. CVPR (2009)
10. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Symposium on Geometry Processing. Eurographics Association (2006)
11. Kolev, K., Klodt, M., Brox, T., Cremers, D.: Continuous global optimization in multiview 3d reconstruction. Int. Journal of Computer Vision 84(1), 80–96 (2009)
12. Lempitsky, V., Boykov, Y.: Global optimization for shape fitting. In: Proc. CVPR (2007)
13. Lu, C.P., Hager, G.D., Mjolsness, E.: Fast and globally convergent pose estimation from video images. IEEE Trans. Pattern Anal. Mach. Intell. 22(6), 610–622 (2000)
14. Newcombe, R., Lovegrove, S., Davison, A.: DTAM: Dense tracking and mapping in Real-Time. In: Proc. ICCV (2011)
15. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of Multi-View stereo reconstruction algorithms. In: Proc. CVPR (2006)
16. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3D. In: ACM Transactions on Graphics (TOG). vol. 25, pp. 835–846. ACM (2006)
17. Steinbruecker, F., Pock, T., Cremers, D.: Large displacement optical flow computation without warping. In: Proc. ICCV (2009)
18. Sundaram, N., Brox, T., Keutzer, K.: Dense point trajectories by gpu-accelerated large displacement optical flow. In: Proc. ECCV. LNCS, Springer (2010)
19. Weickert, J.: Anisotropic diffusion in image processing. B.G. Teubner (1998)
20. Wu, C., Agarwal, S., Curless, B., Seitz, S.: Multicore bundle adjustment. In: Proc. CVPR (2011)
21. Yezzi, A.J., Soatto, S.: Structure from motion for scenes without features. In: Proc. CVPR (2003)
22. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime tv-l 1 optical flow. In: Pattern Recognition. LNCS, vol. 4713, pp. 214–223. Springer (2007)