Parallel Generalized Thresholding Scheme for Live Dense Geometry from a Handheld Camera

Jan Stühmer[1,2], Stefan Gumhold[2], and Daniel Cremers[1]

[1] Department of Computer Science, TU Munich, Germany
[2] Department of Computer Science, TU Dresden, Germany

Abstract. Inspired by recent successes in parallelized optic flow estimation, we propose a variational method which allows to directly estimate dense depth fields from a single hand-held camera in real-time conditions. In particular we show how the central ingredient of the corresponding optic flow method, namely a thresholding scheme, can be generalized to the problem of geometric reconstruction considered in this paper and how it can be parallelized on recent graphics cards. We compare alternative parallelization strategies and experimentally validate that high-quality depth maps can be computed in a few milliseconds from a hand-held camera.

1 Introduction

1.1 From Optic Flow to Geometric Reconstruction

Over the last years parallel algorithms accelerated by means of graphics hardware have revolutionized many areas of Computer Vision, bringing computationally intense challenges within the realm of real-time applications. One of the major breakthroughs in this context was the acceleration of variational optical flow algorithms [1] which allow to compute highly accurate dense motion fields at 640×480 pixels with speeds well above 60 frames per second.

For many computer vision problems the optical flow between two frames provides a correspondence between pairs of pixels in either image which is then further processed, for example to track articulated object models [2] or to reconstruct the depth field of a scene [3]. Yet, in many such cases one is not directly interested in the estimated flow field: For example when reconstructing a static scene from a moving camera as recently done in [3], the estimation of a motion vector field seems entirely unnecessary since apart from the 6-parameter camera motion everything else is static. One may therefore ask: How can we exploit the drastic accelerations of such parallel algorithms without actually computing a flow field?

Recently Stühmer et al. [4] proposed a variational approach to compute dense depth maps from a handheld camera. The estimation of dense geometry from a handheld camera is formulated as a variational approach that can be solved by algorithms that are quite reminiscent of optical flow approaches. Yet rather than computing a vector field that assigns a velocity to each pixel, the geometry

K.N. Kutulakos (Ed.): ECCV 2010 Workshops, Part II, LNCS 6554, pp. 450–462, 2012.

of the scene is directly determined in a coarse-to-fine manner. In particular, the central algorithmic component, namely the thresholding scheme proposed in [1] for computing the primal variables can be generalized to the geometry reconstruction problem. In this paper, we revisit this formulation and show how the arising thresholding scheme can be efficiently implemented on graphics cards.

1.2 Related Work

The reconstruction of dense geometry from images is a major challenge in computer vision. Several methods for stereo reconstruction have been suggested, that compute a disparity map from two images. By using GPU-accelerated algorithms, some of these approaches are even realtime capable, for example those based on belief propagation [5]. More precise and very detailed results can be obtained by using multiple input images [6,7]. Because existing multiview stereo approaches usually require calibrated input images from known camera positions and because of the computational complexity, these methods cannot be used in realtime-applications and therefore have been restricted to offline processing.

Recent developments of keyframe based structure from motion algorithms allow highly accurate camera pose estimation in realtime [8]. However, these approaches represent the scene as a sparse point cloud and do not allow a dense reconstruction of the geometry in front of the camera.

Two early precursors of variational approaches to estimate dense depth maps were proposed in [9,10] One of the central differences of our approach is that it makes use of quadratic relaxation and an efficient primal dual optimization strategy and allows to use robust error norms both for the data term and the regularizer.

The usage of graphics hardware as processing platform for computer vision problems has lead to realtime variational approaches in the field of optic flow computation. By parallelizing the computation on the GPU, even sophisticated PDE methods can be implemented in realtime. Highly accurate dense optic flow can be computed by using a total variation regularizer and a robust L^1-norm error measure for the data term [1]. Because both the regularizer and the data term are not continuously differentiable, the minimization of the energy functional involves some computational difficulties. For the minimization of the L^1-norm data term, a so called thresholding scheme has to be used.

In this paper we will provide a generalization of the thresholding scheme used in optic flow computation, that allows live reconstruction of dense geometry from multiple images. We show in detail how the generalized thresholding scheme can be parallelized and therefore efficiently computed on the GPU. A combination of our method with realtime camera tracking allows live dense geometry reconstruction from the images of a handheld camera.

1.3 Variational Methods for Realtime Optic Flow

Zach et al. [1] suggested the following energy functional for the estimation of dense optic flow

$$E(u) = \int_{\Omega} \left\{ |\nabla u| + \lambda \left| I_1 \left(\mathbf{x} + u(\mathbf{x}) \right) - I_0 \left(\mathbf{x} \right) \right| \right\} d\mathbf{x}, \tag{1}$$

where Ω is the image domain, I_0 and I_1 are two given images and u is the sought vector field that describes the optic flow between both images. The weighting parameter λ controls the influence of the data term in relation to the total variation regularizer.

This functional is not continuously differentiable, and therefore cannot be minimized directly using the Euler-Lagrange formalism. The authors propose to decouple the data term and the regularizer, as it has been previously suggested by Aujol et al. [11]. This leads to the following convex approximation

$$E_\theta = \int_{\Omega} \left\{ |\nabla u| + \frac{1}{2\theta}(u - v)^2 + \lambda \left| \rho(v, \mathbf{x}) \right| \right\} d\mathbf{x}, \tag{2}$$

where $\theta > 0$ is a small constant. With ρ we denote the residual of the linearized data term

$$\rho(v, \mathbf{x}) := I_1(\mathbf{x} + u_0) + \langle \nabla I_1(\mathbf{x} + u_0), v - u_0 \rangle - I_0(\mathbf{x}), \tag{3}$$

where u_0 is a given flow field.

Because the regularizer and data term are decoupled and do not share any variables, a solution of Eq. 2 can be obtained with an alternating minimization scheme. The first step of this alternating scheme is the minimization of Eq. 2 for u. This sub problem is also known as the ROF energy model for image denoising [12] and can be solved using Chambolle's algorithm [13]. By minimizing Eq. 2 for v we obtain the update step of the data term. This update can be computed with a relatively simple thresholding scheme that follows directly from the three possible cases $\rho(v) > 0$, $\rho(v) < 0$ and $\rho(v) = 0$.

By subsequently solving the convex minimization problem and taking each new solution as point u_0 for the linearization of the data term, the flow field can be computed in an iterative warping scheme.

2 Dense Depthmap Estimation from Multiple Images

Instead of estimating a vector field of two-dimensional optic flow vectors, we will provide a method for dense geometry reconstruction from multiple images by minimizing the functional

$$E(h) = \lambda \int_{\Omega_0} \sum_{i \in \mathcal{I}(\mathbf{x})} \left| I_i \left(\pi \left(g_i (h \cdot \mathbf{x}) \right) \right) - I_0 \left(\mathbf{x} \right) \right| d\mathbf{x} + \int_{\Omega_0} |\nabla h| \, d\mathbf{x} \tag{4}$$

with respect to a scalar depth field $h : \Omega \to \mathbb{R}$. Here $\mathbf{x}$ denotes the 2D image location in homogeneous coordinates, $h \cdot \mathbf{x}$ denotes the corresponding 3D coordinate and g_i the rigid body transformation into the camera frame i, and π is the projection from homogeneous coordinates to pixel coordinates regarding a

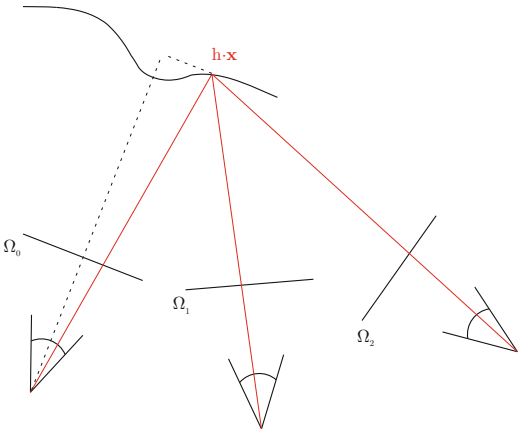


Fig. 1. The depthmap h is defined for the coordinate frame of camera 0. The 3D-point $h \cdot \mathbf{x}$ lies on the surface of the depthmap.

calibrated camera model, in the simplest case this is the perspective projection $\pi\left(x\ y\ z\right) = \left(x/z\ y/z\ 1\right)$.

The set $\mathcal{I}(\mathbf{x})$ contains the indices of all images for which $\pi(g_i(h \cdot \mathbf{x}))$ is inside the image boundaries of I_i. In the following we will use the short form $I_i(h, \mathbf{x})$ for $I_i\left(\pi(g_i(h \cdot \mathbf{x}))\right)$.

This functional is inspired by variational optic flow methods where a robust regularizer allows to preserve discontinuities in the displacement field. By using the L^1 error measure also in the data term, outliers can be handled robustly. In our case we expect similar advantages: The total variation regularizer enables the reconstruction of dense continuous surfaces while preserving discontinuities at object boundaries. The sum of L^1-norm error measures in the data term is motivated by robust statistics and provides robustness against outliers that arise from sensor noise, illumination changes and occlusion. However, using these robust error norms gives rise to some difficulties when solving the functional, that we will address in the following.

We linearize the images I_i by using a first order Taylor expansion, i.e.

$$I_i(h, \mathbf{x}) = I_i(h_0, \mathbf{x}) + (h - h_0)\, \frac{d}{dh} I_i(h, \mathbf{x})\Big|_{h_0} \tag{5}$$

where h_0 is a given depth map. Because this linearization only holds for small innovations of the depthmap, the whole minimization process is embedded into a coarse-to-fine warping strategy [14,15].

The derivative $\frac{d}{dh} I_i(h, \mathbf{x})$ can be considered as a directional derivative in direction of a differential vector on the image plane of I_i that results from a variation

of h. By using the chain rule, this derivative can be expressed as the scalar product of the gradient of $I_i(h, \mathbf{x})$ with the mentioned differential vector, i.e.

$$\frac{d}{dh} I_i(h, \mathbf{x}) = \nabla I_i(h, \mathbf{x}) \cdot \frac{d}{dh} \pi(g_i(h \cdot \mathbf{x})). \tag{6}$$

The differential vector

$$\frac{d}{dh} \pi(g_i(h \cdot \mathbf{x})) = \begin{pmatrix} \frac{d}{dh} x' \\ \frac{d}{dh} y' \end{pmatrix} \tag{7}$$

needs to be computed with respect to the chosen camera model.

With above linear approximation for $I_i(h, \mathbf{x})$ we can express the current residual of the data term for input image i as

$$\rho_i(h, \mathbf{x}) := I_i(h_0, \mathbf{x}) + (h - h_0) \frac{d}{dh} I_i(h, \mathbf{x}) \Big|_{h_0} - I_0(\mathbf{x}) \tag{8}$$

Inserting this expression into the original energy functional (Eq. 4) gives

$$E(h) = \lambda \int_\Omega \sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h, \mathbf{x})| \, d\mathbf{x} + \int_\Omega |\nabla h| \, d\mathbf{x}. \tag{9}$$

This functional is still difficult to minimize, because it is not continuously differentiable and therefore the Euler-Lagrange formalism cannot be used directly. By decoupling the data term and the regularizer [11] we get the following convex approximation of Eq. 4:

$$E_\theta = \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - h)^2 + \lambda \sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h, \mathbf{x})| \right\} d\mathbf{x}. \tag{10}$$

The proposed approximation Eq. 10 is convex, thus the functional can be minimized using an alternating minimization procedure in u and h:

1. For fixed h solve Eq. 10 for u

$$\min_u \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u - h)^2 \right\} d\mathbf{x}. \tag{11}$$

   This optimization problem is exactly the ROF model [12], with θ as regularization parameter. We can use Chambolle's projected gradient descend method to solve this problem [13].

2. For fixed u solve Eq. 10 for h

$$\min_h \int_\Omega \left\{ \frac{1}{2\theta}(u - h)^2 + \lambda \sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h, \mathbf{x})| \right\} d\mathbf{x}. \tag{12}$$

   This minimization problem can be solved point-wise, because it does not depend on any spatial derivatives of u any more. We will show in the following, how this minimization problem can be solved efficiently with a generalized thresholding scheme.

3 Generalized Thresholding Scheme

The second step of the alternation scheme offers some difficulties, because the sum of absolute valued functions results in multiple critical points, where the whole data term is not differentiable. Thus, a simple thresholding scheme as in the optical flow problem cannot be used. Nevertheless we will provide a generalization of the thresholding scheme that allows a closed-form solution of Eq. 12, that is a further generalization of the concept presented in [7].

For fixed h_0 and $\mathbf{x}$ the linearized data term ρ_i for each image Eq. 8 can be written in the general form of a linear function

$$\rho_i(h, \mathbf{x}) = a_i\, h + b_i, \tag{13}$$

where

$$a_i := I_i^h(\mathbf{x}) \quad \text{and} \quad b_i := I_i(h, \mathbf{x}_0) - h_0\, I_i^h(\mathbf{x}) - I_0(\mathbf{x}). \tag{14}$$

In the following we will consider h_0 and $\mathbf{x}$ as fixed and therefore we simplify our notation and omit the dependencies of a_i and b_i from these fixed values.

The absolute valued functions $|\rho_i(h)|$ are differentiable with respect to h except at their critical points, where one of the ρ_i equals zero and changes its sign. Let us denote these critical points as

$$t_i := -\frac{b_i}{a_i} = -\frac{I_i(h, \mathbf{x}_0) - h_0\, I_i^h(\mathbf{x}) - I_0(\mathbf{x})}{I_i^h(\mathbf{x})}\,, \tag{15}$$

where $i \in \mathcal{I}(\mathbf{x})$.

At these points Eq. 11 is not differentiable, as the corresponding ρ_i changes its sign. Without loss of generality we can assume that $t_i \leq t_{i+1}$, i.e. we obtain a sorted sequence of $\{\rho_i : i \in \mathcal{I}(\mathbf{x})\}$, that is sorted by the values of their critical points. In order to avoid special cases we add $t_0 = -\infty$ and $t_{|\mathcal{I}(\mathbf{x})|+1} = +\infty$ to this sequence.

Proposition 1. *The minimizer of Eq. 12 can be found using the following strategy: If the stationary point*

$$h_1 := u - \lambda\theta \left(\sum_{i \in \mathcal{I}(\mathbf{x}) : i \leq k} I_i^h(\mathbf{x}) - \sum_{j \in \mathcal{I}(\mathbf{x}) : j > k} I_j^h(\mathbf{x}) \right) \tag{16}$$

lies in the interior of (t_k, t_{k+1}) for some $k \in \mathcal{I}(\mathbf{x})$, then $h = h_1$. Else the minimizer of Eq. 12 can be found among the set of critical points:

$$h = \arg\min_{h_2 \in \{t_i\}} \left(\frac{1}{2\theta}(u - h_2)^2 + \lambda \sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h_2, \mathbf{x})| \right). \tag{17}$$

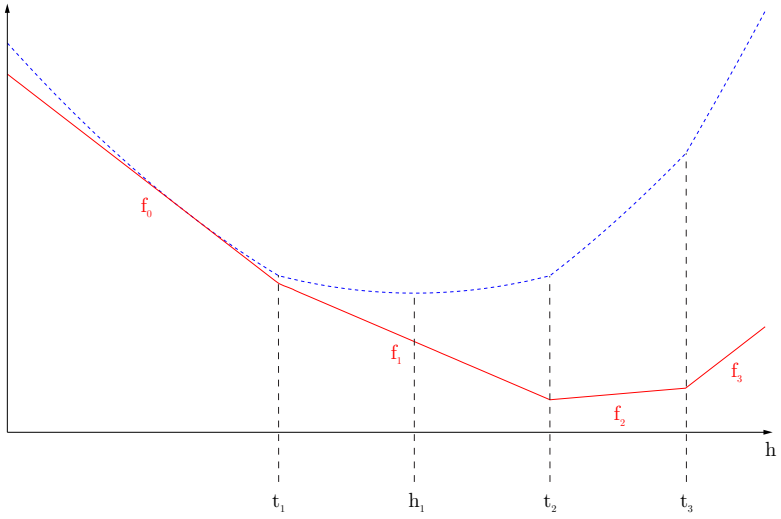


Fig. 2. The minimizations problem in the second step of the alternation scheme Eq. 12 (blue) can be written as the sum of a quadratic function and a piecewise linear function (red). In the interior of the intervals (t_k, t_{k+1}) this term is differentiable with respect to h. In this illustration the minimum is at the critical point h_1 that lies in the interval (t_1, t_2).

Proof. First we show, that Eq. 12 can be written as the sum of a quadratic function with a linear function in the interior of each interval (t_k, t_{k+1}) with $k \in \{0, |\mathcal{I}(\mathbf{x})|\}$. This is also illustrated in Fig. 2. We replace the absolute value function by using the signum function

$$\sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h, \mathbf{x})| = \sum_{i \in \mathcal{I}(\mathbf{x})} |a_i\, h + b_i| \tag{18}$$

$$= \sum_{i \in \mathcal{I}(\mathbf{x})} \Big\{ \operatorname{sgn}\left(\rho_i(h, \mathbf{x})\right)\ (a_i\, h + b_i) \Big\} \tag{19}$$

$$= \sum_{i \in \mathcal{I}(\mathbf{x})} \Big\{ \operatorname{sgn}\left(\rho_i(h, \mathbf{x})\right)\ a_i \Big\}\, h + \sum_{i \in \mathcal{I}(\mathbf{x})} \Big\{ \operatorname{sgn}\left(\rho_i(h, \mathbf{x})\right)\ b_i \Big\}. \tag{20}$$

In order to write above equation in the general form of linear functions, we need to eliminate the signum functions. Let us consider the interior of an interval (t_k, t_{k+1}) with $k \in \{0, |\mathcal{I}(\mathbf{x})|\}$. If h' lies in the interior of the interval (t_k, t_{k+1}), i.e. $h' > t_k$ and $h' < t_{k+1}$, then by definition of the sorted sequence $\{\rho_i\}$ it holds that

$$\operatorname{sgn}\left(\rho_i(h', \mathbf{x})\right) = +1 \text{ if } i < k \tag{21}$$

$$\operatorname{sgn}\left(\rho_i(h', \mathbf{x})\right) = -1 \text{ if } i \geq k. \tag{22}$$

By replacing the signum functions with above expressions the data term can be written in the general form of a linear function f_k for each interval $k \in \{0, |\mathcal{I}(\mathbf{x})|\}$

$$\sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(h', \mathbf{x})| = \tilde{a}_k \, h' + \tilde{b}_k =: f_k(h') \tag{23}$$

with

$$\tilde{a}_k = \sum_{i \in \mathcal{I}(\mathbf{x}) : i < k} a_i - \sum_{j \in \mathcal{I}(\mathbf{x}) : j \geq k} a_j \tag{24}$$

and

$$\tilde{b}_k = \sum_{i \in \mathcal{I}(\mathbf{x}) : i < k} b_i - \sum_{j \in \mathcal{I}(\mathbf{x}) : j \geq k} b_j. \tag{25}$$

As a result Eq. 12 can be written as

$$\frac{1}{2\theta}(u - h')^2 + \lambda \, f_k(h'), \tag{26}$$

where h' lies in the interior of (t_k, t_{k+1}).

By differentiating above equation with respect to h' we get the stationary point

$$h_1 = u - \lambda \, \theta \, \tilde{a}_k \tag{27}$$

$$= u - \lambda \, \theta \left(\sum_{i \in \mathcal{I}(\mathbf{x}) : i < k} a_i - \sum_{j \in \mathcal{I}(\mathbf{x}) : j \geq k} a_j \right) \tag{28}$$

$$= u - \lambda \, \theta \left(\sum_{i \in \mathcal{I}(\mathbf{x}) : i \leq k} I_i^h(\mathbf{x}) - \sum_{j \in \mathcal{I}(\mathbf{x}) : j > k} I_j^h(\mathbf{x}) \right) \tag{29}$$

Such a stationary point h_1 exists, if it stays inside the interval (t_k, t_{k+1}) for some $k \in \{0, |\mathcal{I}(\mathbf{x})|\}$. If no stationary point can be found for any of the intervals, the minimizer of Eq. 12 resides on the boundary of one of the intervals, i.e. the minimizer can be found among the set of critical points $\{t_i\}$. □

4 Generalized Thresholding Scheme on the GPU

We implemented the proposed method on the GPU using the CUDA (Compute Unified Device Architecture) framework. For the computation of the generalized thresholding scheme first we need a sequence of the coefficients of ρ_i, that is sorted by the critical points t_i. This sorting operation needs to be performed only once for each linearization of the data term, because the values of t_i and the coefficients do not depend on the further iterations. This step can be computed in a parallel-sequential manner on the graphics hardware, i.e. for each pixel of the depthmap, one thread (x, y) sorts the coefficients of all ρ_i at this point (x, y). Because the number of images is rather small a simple bubblesort algorithm is used in each thread.

The output of this sorting step is the sorted sequence of critical points t_k, the sum of the derivatives

$$e_k := \sum_{i \in \mathcal{I}(\mathbf{x}) : i \leq k} I_i^h(\mathbf{x}) - \sum_{j \in \mathcal{I}(\mathbf{x}) : j > k} I_j^h(\mathbf{x}) \,,$$

and

$$f_k := \sum_{i \in \mathcal{I}(\mathbf{x})} |\rho_i(\mathbf{x}, t_k)| \,, \tag{30}$$

the residuals of the data terms at the critical points. With these coefficients the values of h_1 (Eq. 16) and h_2 (Eq. 17) can be computed efficiently in every iteration of the minimization scheme.

5 Experimental Results

We evaluated two different GPU implementations of the general thresholding scheme. The first (implementation A) is a parallel sequential implementation, where one thread is assigned to each pixel of the depthmap. This thread takes the data of the views $I_1 \dots I_n$ as input and iteratively determines the minimizer by sequential processing of the data of each view.

Because each thread can stop any further computation when the first stationary point is found, the amount of computation varies for each thread. This

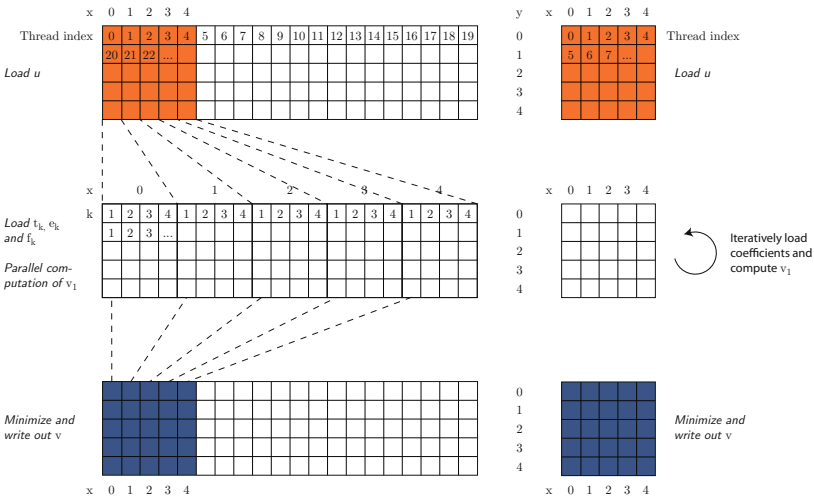
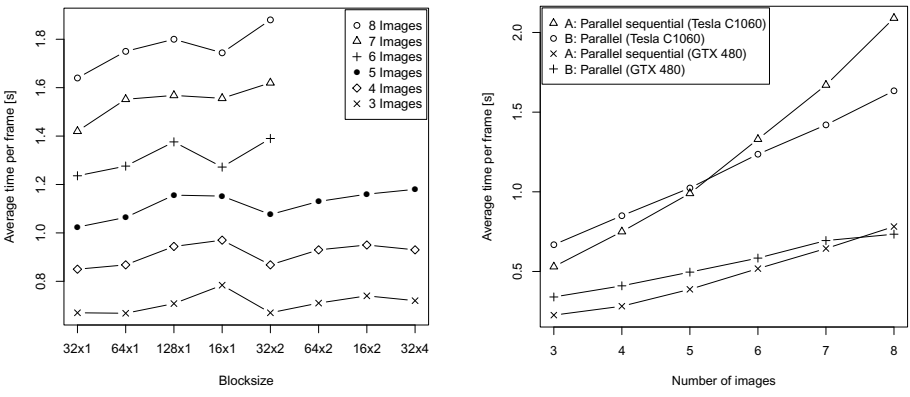


Fig. 3. Two different implementations of the generalized thresholding scheme. The implementation on the left is higher parallelized and allows a better performance balancing when the number of views increases. In the implementation on the right each thread processes the data sequentially.

usually results in suboptimal performance and lower occupancy of the GPU. Therefore we evaluated a second implementation, that is highly parallelized and has a deterministic computational load for each thread. By assigning multiple threads to each pixel of the depthmap, the data of all views I_i can be processed in parallel. While all necessary computations to find the stationary points can be performed highly parallelized, only the last step, the determination of the minimal value, involves sequential processing. In the following we will refer to this kind of implementation as implementation B. The difference of both implementations is depicted in Fig. 3.

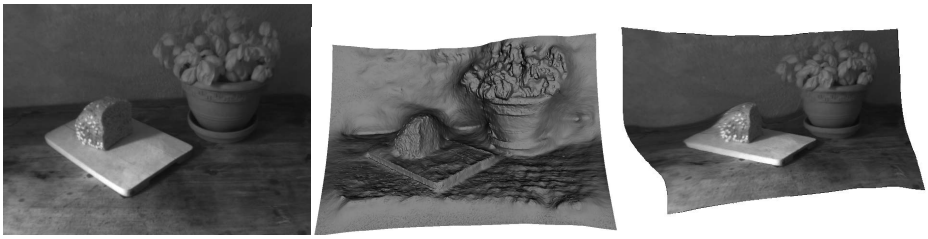
5.1 Comparison of Different GPU Implementations

We optimized the block sizes for both algorithms by searching values of the power of two for the block-width and -height. For the parallelized implementation A, we expected that the optimal block-size would depend on the amount of data that is processed in parallel, in this case on the number of input images. The results show, that the optimal block-size is determined by the size of the small block in Fig. 3, that contains the pixels of the depthmap. While the size of the bigger block where the values of h_1 are actually computed depends on the number of images, the optimal size of the small block stays constant. The dependency between runtime performance and the size of this small block is shown in Fig. 4a for different number of input images. A size of 32×1 outperforms all other tested configurations on a NVidia Tesla C1060. On a recent GTX 480 the optimal size is 64×1.



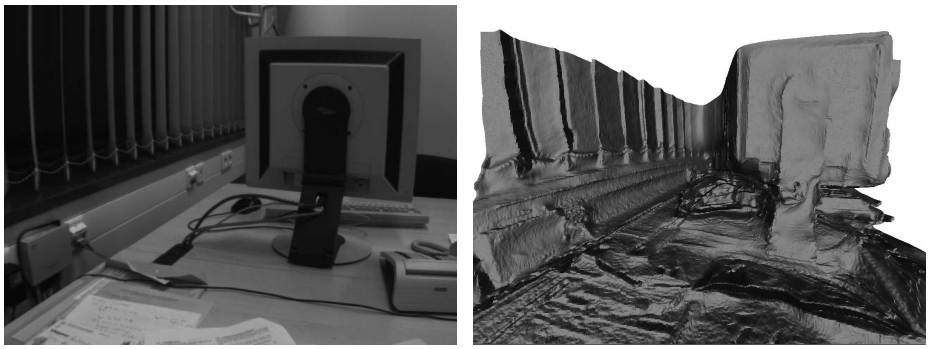
(a) (b)

Fig. 4. *(a)* Performance of algorithm B on a Tesla C1060 for different sizes of the part of the depthmap that is processed in parallel. *(b)* Comparison of both implementation strategies for different number of input images. When the amount of input data increases, the higher parallelized algorithm B is faster than algorithm A.



(a) Reference camera image (b) Reconstructed geometry (c) Synthesized View

Fig. 5. Dense depthmaps estimated from a single moving camera



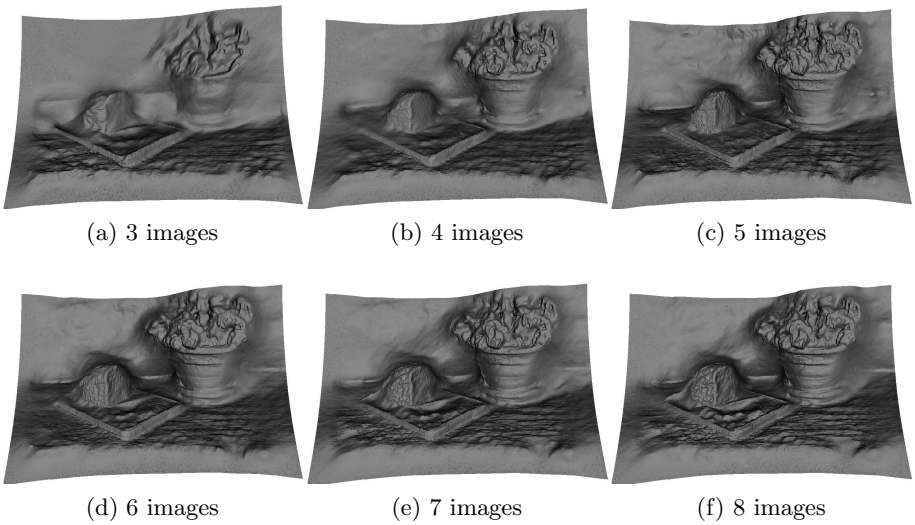
(a) Reference camera image (b) Reconstructed geometry

Fig. 6. Note the accurate reconstruction of small-scale details like the network socket and cords.

We also compared the performance of both implementations for different number of input images. While the parallel sequential implementation A is faster for smaller number of input images, the parallelized implementation B shows a better performance for higher numbers. The exact number of input images, for which implementation B performs better than A, depends on the specific hardware configuration. The parallelized algorithm B shows a linear dependency between runtime performance and the number if input images. The images and the reconstructed depthmap in both experiments are of size 450×375.

5.2 Real World Data

The combination with a recently proposed method for realtime camera tracking [8] allows the reconstruction of dense geometry with a single hand-held camera. Figure 5 shows the input image of the reference camera, the reconstructed geometry and a synthesized view. Another example is shown in figure 6. The proposed method computes a dense geometry rather than the location of sparse



(a) 3 images (b) 4 images (c) 5 images

(d) 6 images (e) 7 images (f) 8 images

Fig. 7. Multiple images allow a more detailed reconstruction.

feature points. Increasing the number of input images allows a finer and more detailed reconstruction as shown in figure 7.

6 Conclusion

In this paper we adapted state-of-the-art variational optic flow algorithms so as to directly generate dense depth maps in a coarse-to-fine primal dual algorithm. The algorithm runs on a single GPU and allows to compute highly accurate dense geometric information within fractions of a second. In particular, we present a GPU implementation of the generalized thresholding scheme arising in the computation of the primal variables. We experimentally compare two alternative strategies of parallelization that differ with respect to the amount of balancing assured across different threads. Experimental results show that one implementation shows a higher performance when the number of views is rather small, while the other strategy is better suited when the input data increases. Highly accurate and detailed results from real world image data are presented.

References

1. Zach, C., Pock, T., Bischof, H.: A Duality Based Approach for Realtime TV-L1 Optical Flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 214–223. Springer, Heidelberg (2007)
2. Brox, T., Rosenhahn, B., Gall, J., Cremers, D.: Combined region- and motion-based 3d tracking of rigid and articulated objects. IEEE Transactions on Pattern Analysis and Machine Intelligence (2009)

3. Newcombe, R.A., Davison, A.J.: Live dense reconstruction with a single moving camera. In: Int. Conf. on Computer Vision and Pattern Recognition (2010)
4. Stühmer, J., Gumhold, S., Cremers, D.: Real-Time Dense Geometry from a Handheld Camera. In: Goesele, M., Roth, S., Kuijper, A., Schiele, B., Schindler, K. (eds.) DAGM 2010. LNCS, vol. 6376, pp. 11–20. Springer, Heidelberg (2010)
5. Yang, Q., Wang, L., Yang, R., Wang, S., Liao, M., Nistér, D.: Real-time global stereo matching using hierarchical belief propagation. In: British Machine Vision Association, BMVC, pp. 989–998. (2006)
6. Kolev, K., Cremers, D.: Continuous ratio optimization via convex relaxation with applications to multiview 3d reconstruction. In: Int. Conf. on Computer Vision and Pattern Recognition, pp. 1858–1864 (2009)
7. Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust TV-L^1 range image integration. In: IEEE Int. Conf. on Computer Vision, Rio de Janeiro, Brazil. LNCS, IEEE (2007)
8. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2007), Nara, Japan (2007)
9. Robert, L., Deriche, R., Faugeras, O.D.: Dense depth recovery from stereo images. In: ECAI 1992: Proceedings of the 10th European Conference on Artificial Intelligence, pp. 821–823. John Wiley & Sons, Inc., New York (1992)
10. Robert, L., Deriche, R.: Dense Depth Map Reconstruction: A Minimization and Regularization Approach which Preserves Discontinuities. In: Buxton, B.F., Cipolla, R. (eds.) ECCV 1996. LNCS, vol. 1064, pp. 439–451. Springer, Heidelberg (1996)
11. Aujol, J.F., Gilboa, G., Chan, T., Osher, S.: Structure-texture image decomposition–modeling, algorithms, and parameter selection. Int. J. Comput. Vision 67, 111–136 (2006)
12. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268 (1992)
13. Chambolle, A.: An algorithm for total variation minimization and applications. J. Math. Im. Vis. 20, 89–97 (2004)
14. Nagel, H., Enkelmann, W.: An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. IEEE Trans. on Patt. Anal. and Mach. Intell. 8, 565–593 (1986)
15. Black, M.J., Anandan, P.: The robust estimation of multiple motions: Parametric and piecewise–smooth flow fields. Comp. Vis. Graph. Image Proc.: IU 63, 75–104 (1996)