Universität des Saarlandes

Fachrichtung 6.1 – Mathematik

Preprint Nr. 89

Variational Optic Flow Computation in Real-Time

Andrés Bruhn, Joachim Weickert, Christian Feddern,
Timo Kohlberger and Christoph Schnörr

Saarbrücken 2003

Fachrichtung 6.1 – Mathematik
Universität des Saarlandes

Preprint No. 89
submitted: June 8, 2003

Variational Optic Flow Computation in Real-Time

Andrés Bruhn
Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science
Saarland University, Building 27.1
66041 Saarbrücken, Germany
bruhn@mia.uni-saarland.de

Joachim Weickert
Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science
Saarland University, Building 27.1
66041 Saarbrücken, Germany
weickert@mia.uni-saarland.de

Christian Feddern
Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science
Saarland University, Building 27.1
66041 Saarbrücken, Germany
feddern@mia.uni-saarland.de

Timo Kohlberger
Computer Vision, Graphics and Pattern Recognition Group
Dept. of Mathematics and Computer Science
University of Mannheim
68131 Mannheim, Germany
kohlberger@uni-mannheim.de

Christoph Schnörr
Computer Vision, Graphics and Pattern Recognition Group
Dept. of Mathematics and Computer Science
University of Mannheim
68131 Mannheim, Germany
schnoerr@uni-mannheim.de

Edited by
FR 6.1 – Mathematik
Universität des Saarlandes
Postfach 15 11 50
66041 Saarbrücken
Germany

Fax: + 49 681 302 4443
e-Mail: preprint@math.uni-sb.de
WWW: http://www.math.uni-sb.de/

Abstract

Variational methods for optic flow computation have the reputation of producing good results at the expense of being too slow for real-time applications. We show that real-time variational computation of optic flow fields is possible when appropriate methods are combined with modern numerical techniques. We consider the CLG method, a recent variational technique that combines the quality of the dense flow fields of the Horn and Schunck approach with the noise robustness of the Lucas–Kanade method. For the linear system of equations resulting from the discretised Euler–Lagrange equations, we present different multigrid schemes in detail. We show that under realistic accuracy requirements they are up to 247 times more efficient than the widely used Gauß-Seidel algorithm. On a 3.06 GHz PC, we have computed 40 dense flow fields of size 200×200 pixels within a single second.

AMS 2000 Subject Classification: 68T45, 65N55, 49K20, 65K10, 35J60, 65N04
Key Words: computer vision, optic flow, differential techniques, variational methods, multigrid methods, structure tensor, partial differential equations.

Contents

1 Introduction

Variational methods belong to the well-established techniques for estimating the displacement field *(optic flow)* in an image sequence. They perform well in terms of different error measures [2, 12] and they make all model assumptions explicit in a transparent way. Moreover, they yield dense flow fields, and it is straightforward to derive continuous models that are rotationally invariant. These properties make continuous variational models appealing for a number of applications. For a survey of these techniques we refer to [26].

Variational methods, however, require the minimisation of a suitable energy functional. Often this is achieved by discretising the associated Euler–Langrange equations and solving the resulting systems of equations in an iterative way. Classical iterative methods such as Jacobi or Gauß–Seidel iterations are frequently applied [28]. In this case one observes that the convergence is reasonably fast in the beginning, but after a while it deteriorates significantly such that often several thousands of iterations are needed in order to obtain the required accuracy. As a consequence, variational optic flow methods are usually considered to be too slow when real-time performance is needed.

The goal of the present paper is to show that it is possible to make variational optic flow methods suitable for real-time applications by combining them with state-of-the-art numerical techniques. We use the recently introduced CLG method [8, 25], a variational technique that combines the advantages of two classical optic flow algorithms: the variational Horn and Schunck approach [16], and the local least-square technique of Lucas and Kanade [17]. For the CLG method we derive fast numerical schemes based on so-called multigrid strategies [5, 6, 15, 24, 27]. Such techniques belong to the fastest numerical methods for solving linear systems of equations. We present our algorithms in detail and show that an appropriate Full Multigrid scheme leads to a speed-up of more than two orders of magnitude compared to widely used iterative methods. As a consequence, it becomes possible to compute 40 optic flow frames per second on a standard PC, when image sequences of size 200×200 pixels are used.

Our paper is organised as follows. In Section 2 we review the CLG model as a representative for variational optic flow methods. Section 3 shows how this problem can be discretised. Fast multigrid strategies for the CLG approach are derived in Section 4. In Section 5 we compare this algorithm with the widely used Gauß–Seidel and SOR schemes and show that it allows real-time computation of optic flow. The paper is concluded with a summary in Section 6.

Related work. It is quite common to use pyramid strategies for speeding up variational optic flow methods; see e.g. [1, 18, 19]. They use the solution at a coarse grid as initialisation on the next finer grid. Such techniques may be regarded as the simplest multigrid strategy, namely cascading multigrid. The mathematical literature on multigrid methods shows that one can expect improved performance by using more advanced multigrid strategies. Such techniques, however, are not used very frequently in the context of variational optic flow computation. First proposals go back to Glazer [14], Terzopoulos [23] and Enkelmann [10]. More recently, Ghosal and Vaněk [13] developed an algebraic multigrid method for an anisotropic variational approach that can be related to Nagel's method [20]. Zini et al. [29] proposed a conjugate gradient-based multigrid technique for an extension of the Horn and Schunck functional, and Borzi et al. [4] investigated a full approximation scheme (FAS) for a control formulation of a nonlinear optic flow problem. To the best of our knowledge, our paper is the first work that reports real-time performance for variational optic flow techniques on standard hardware. A preliminary shorter version of the present paper has been presented at a conference [7].

2 Optic Flow Computation with the CLG Approach

In [8, 25] we have introduced the so-called *combined local-global (CLG) method* for optic flow computation. It combines the advantages of the global Horn and Schunck approach [16] and the local Lucas–Kanade method [17]. In order to describe the underlying idea behind the CLG method, let $f(x, y, t)$ be an image sequence, where (x, y) denotes the location within a rectangular image domain Ω and t is the time. The CLG method computes the optic flow field $(u(x, y), v(x, y))^\top$ at some time t as the minimiser of the energy functional

$$E(u, v) = \int_\Omega \left(w^\top J_\rho(\nabla_3 f)\, w + \alpha(|\nabla u|^2 + |\nabla v|^2) \right) dx\, dy, \tag{1}$$

where the vector field $w(x, y) = (u(x, y), v(x, y), 1)^\top$ describes the displacement, ∇u is the spatial gradient $(u_x, u_y)^\top$, and $\nabla_3 f$ denotes the spatiotemporal gradient $(f_x, f_y, f_t)^\top$. The matrix $J_\rho(\nabla_3 f)$ is the structure tensor [3, 11, 22] given by $K_\rho * (\nabla_3 f\, \nabla_3 f^\top)$, where $*$ denotes convolution, and K_ρ is a Gaussian with standard deviation ρ. The weight $\alpha > 0$ serves as regularisation parameter.

For $\rho \to 0$ the CLG approach comes down to the Horn and Schunck method, and for $\alpha \to 0$ it becomes the Lucas–Kanade algorithm. It combines the dense

flow fields of Horn–Schunck with the high noise robustness of Lucas–Kanade. For a detailed performance evaluation we refer to [8, 25].
In order to recover the optic flow field, the energy functional $E(u, v)$ has to be minimised. As is well-known from the calculus of variations [9], this can be done by solving its Euler–Lagrange equations

$$\alpha\Delta u - (J_{11}(\nabla_3 f)\, u + J_{12}(\nabla_3 f)\, v + J_{13}(\nabla_3 f)) = 0, \tag{2}$$

$$\alpha\Delta v - (J_{12}(\nabla_3 f)\, u + J_{22}(\nabla_3 f)\, v + J_{23}(\nabla_3 f)) = 0, \tag{3}$$

where Δ denotes the Laplacean, and reflecting boundary conditions are applied, i.e. the normal derivative vanishes at boundaries:

$$\partial_n u = 0, \qquad \partial_n v = 0 \qquad \text{on } \partial\Omega. \tag{4}$$

3 Discretisation

Now we are in a position to investigate typical algorithms for solving the Euler-Lagrange equations numerically. First we present a finite difference discretisation, and then we discuss two algorithms for solving the resulting linear systems of equations.

3.1 The Discrete Euler-Lagrange Equations

Let us now investigate a suitable discretisation for the CLG method (2)–(3). To this end we consider the unknown functions $u(x, y, t)$ and $v(x, y, t)$ on a rectangular pixel grid of cell size $h_x \times h_y$, and we denote by u_i the approximation to u at some pixel i with $i = 1,...,N$. Gaussian convolution is realised by discrete convolution with a truncated and renormalised Gaussian, where the truncation took place at 3 times the standard deviation. Symmetry and separability have been exploited in order to speed up these discrete convolutions. Spatial derivatives of the image data f have been approximated using central differences, while temporal derivatives are approximated with a simple two-point stencil. Let us denote by J_{nmi} the component (n, m) of the structure tensor $J_\rho(\nabla f)$ in some pixel i. Furthermore, let $\mathcal{N}_l(i)$ denote the set of neighbours of pixel i in direction of axis l. Then a finite difference approximation to the Euler–Lagrange equations (2)–(3) with reflecting

boundary conditions is given by

$$0 = \alpha \sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{u_j - u_i}{h_l^2} \; - \; \left(J_{11i}\, u_i + J_{12i}\, v_i + J_{13i} \right), \tag{5}$$

$$0 = \alpha \sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{v_j - v_i}{h_l^2} \; - \; \left(J_{21i}\, u_i + J_{22i}\, v_i + J_{23i} \right) \tag{6}$$

for $i = 1,...,N$. This constitutes a linear system of equations for the $2N$ unknowns (u_i) and (v_i).

3.2 The Gauß–Seidel Method

The preceding linear system of equations has a sparse system matrix. It may be solved iteratively, e.g. by applying the Gauß–Seidel method [28]. Because of its simplicity it is frequently used in literature. If the upper index k denotes the iteration step, the Gauß-Seidel method can be written as

$$u_i^{k+1} = \frac{\sum\limits_{l=1}^{2} \frac{\alpha}{h_l^2} \Big(\sum\limits_{j \in \mathcal{N}_l^-(i)} u_j^{k+1} + \sum\limits_{j \in \mathcal{N}_l^+(i)} u_j^k \Big) - \left(J_{12i}\, v_i^k + J_{13i} \right)}{\sum\limits_{l=1}^{2} \frac{\alpha}{h_l^2} |\mathcal{N}_l(i)| + J_{11i}}, \tag{7}$$

$$v_i^{k+1} = \frac{\sum\limits_{l=1}^{2} \frac{\alpha}{h_l^2} \Big(\sum\limits_{j \in \mathcal{N}_l^-(i)} v_j^{k+1} + \sum\limits_{j \in \mathcal{N}_l^+(i)} v_j^k \Big) - \left(J_{21i}\, u_i^{k+1} + J_{13i} \right)}{\sum\limits_{l=1}^{2} \frac{\alpha}{h_l^2} |\mathcal{N}_l(i)| + J_{22i}} \tag{8}$$

where $\mathcal{N}_l^-(i) := \{ j \in \mathcal{N}_l(i) \mid j < i \}$ and $\mathcal{N}_l^+(i) := \{ j \in \mathcal{N}_l(i) \mid j > i \}$. By $|\mathcal{N}_l(i)|$ we denote the number of neighbours of pixel i in direction of axis l that belong to the image domain.

3.3 The Pointwise Coupled Gauß–Seidel Method

Although u_i^{k+1} and v_i^{k+1} are computed sequentially at each pixel, this form of coupling between the two equations is rather weak. Particularly with regard to the increasing influence of the coupled zeroth order terms for decreasing

values of α, one may think of a simultaneous solving for both unknowns instead. Such a proceeding yields the so called *pointwise coupled Gauß-Seidel* solver, a block Gauß-Seidel variant, where all unknowns at a pixel are updated synchroneously. If w_i^{k+1} denotes the flow vector $(u_i^{k+1}, v_i^{k+1})^\top$ at pixel i, the pointwise coupled Gauß-Seidel method can be written as

$$w_i^{k+1} = M_i^{-1} g_i^{k+1/2} \tag{9}$$

with

$$M_i := \begin{pmatrix} \sum_{l=1}^{2} \frac{\alpha}{h_l^2} |\mathcal{N}_l(i)| + J_{11i} & J_{12i} \\ J_{21i} & \sum_{l=1}^{2} \frac{\alpha}{h_l^2} |\mathcal{N}_l(i)| + J_{22i} \end{pmatrix}, \tag{10}$$

and

$$g_i^{k+1/2} := \begin{pmatrix} \sum_{l=1}^{2} \frac{\alpha}{h_l^2} \Big(\sum_{j \in \mathcal{N}_l^-(i)} u_j^{k+1} + \sum_{j \in \mathcal{N}_l^+(i)} u_j^k \Big) - J_{13i} \\ \sum_{l=1}^{2} \frac{\alpha}{h_l^2} \Big(\sum_{j \in \mathcal{N}_l^-(i)} v_j^{k+1} + \sum_{j \in \mathcal{N}_l^+(i)} v_j^k \Big) - J_{23i} \end{pmatrix}. \tag{11}$$

One step of the pointwise coupled Gauß–Seidel method is computationally more expensive than the usual Gauß–Seidel method, since it requires to solve a 2×2 linear system in each pixel. In Section 5, however, we will see that this additional effort may pay off in certain situations.

4 An Efficient Multigrid Algorithm

4.1 Motivation

Common iterative solvers like the two presented Gauß–Seidel variants usually perform very well in removing the higher frequency parts of the error within the first iterations. This behaviour is reflected in a good initial convergence rate. Because of their smoothing properties regarding the error, these solvers are referred to as *smoothers*. After some iterations, however, only low frequency components of the error remain and the convergence slows down

significantly. At this point smoothers suffer from their local design and cannot attenuate efficiently low frequencies that have a sufficient large wave length in the spatial domain.
Multigrid methods [5, 6, 15, 24, 27] overcome this problem by creating a sophisticated fine-to-coarse hierarchy of equation systems with excellent error reduction properties. Low frequencies on the finest grid reappear as higher frequencies on coarser grids, where they can be removed successfully. This strategy allows multigrid methods to compute accurate results much faster than non–hierarchical iterative solvers. Since we focus on the real-time computation of optic flow, we have developed such multigrid algorithms for the CLG approach.

4.2 Detailed Description of a 2-Grid Cycle

Let us now explain our strategy in detail. We reformulate the linear system of equations given by (5)–(6) as

$$A^h x^h = f^h \tag{12}$$

where the index h stands for the grid cell size $h_x \times h_y$, x^h is the concatenated vector $((u^h)^\top, (v^h)^\top)^\top$, f^h is the right hand side given by $\frac{1}{\alpha}((J_{13}^h)^\top, (J_{23}^h)^\top)^\top$ and A^h is a symmetric positive definite matrix. Let $\tilde{x}^h$ be the result computed by the chosen Gauß–Seidel smoother after n_1 iterations. Then the error of the solution is given by

$$e^h = x^h - \tilde{x}^h. \tag{13}$$

Evidently, one is interested in finding e^h in order to correct the approximated solution $\tilde{x}^h$. Since the error cannot be computed directly, we determine the residual error given by

$$r^h = f^h - A^h \tilde{x}^h \tag{14}$$

instead. Since A is a linear operator, we have

$$A^h e^h = r^h. \tag{15}$$

Solving this system of equations would give us the desired correction e^h. Since high frequencies of the error have already been removed by our smoother , we can solve this system at a coarser level. For the sake of clarity the notation for the coarser grid is chosen correspondingly to the original equation on the fine grid (12). Thus, the linear equation system (15) becomes

$$A^H x^H = f^H \tag{16}$$

at the coarser level, where H is the index for the new cell size $H_x \times H_y$ with $H_x \geq h_x$ and $H_y \geq h_y$. The right hand side f^H is a downsampled version of r^h.

At this point we have to make four decisions:

(I) The *cell size* H on the coarser grid has to be chosen. In our implementation $H_x \times H_y$ is computed as follows : Let N_x^h and N_y^h be the number of cells on the fine grid in x and y direction. Then the new cell size is given by

$$H_x := h_x \frac{N_x^h}{N_x^H} \qquad\qquad H_y := h_y \frac{N_y^h}{N_y^H} \tag{17}$$

with $N_x^H = \lceil N_x^h/2 \rceil$ and $N_y^H = \lceil N_y^h/2 \rceil$, where $\lceil z \rceil$ denotes the smallest integer number m with $m \geq z$. Thus the total number of cells at the coarser grid is $N^H = N_x^H N_y^H$.

(II) A *restriction operator* $R^{h \to H}$ has to be defined that allows the transfer of vectors from the fine to the coarse grid. By its application to the residual r^h we obtain the right hand side of the equation system on the coarser grid:

$$f^H = R^{h \to H} r^h. \tag{18}$$

For simplicity, averaging over $H_x \times H_y$ is used for $R^{h \to H}$ (see Fig. 1).

(III) A *coarser version of matrix* A^h has to be created. All entries of A^h belonging to the discretised Laplacean depend on the grid cell size of the solution x^h. In order to obtain the corresponding coarse grid entries, a simple adapation of the grid size is sufficient:

$$\sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{u_i^H - u_j^H}{H_l^2}. \tag{19}$$

Having their origin in the structure tensor J^h, all other entries of A^h are independent of x^h. Hence their coarse grid counterparts are obtained by a componentwise restriction of J^h:

$$J_{nm}^H = R^{h \to H} J_{nm}^h. \tag{20}$$

This allows the formulation of the coarse grid equation system

$$0 = \alpha \sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{u_j^H - u_i^H}{H_l^2} \; - \; \left(J_{11i}^H \, u_i^H + J_{12i}^H \, v_i^H + f_{1i}^H \right), \tag{21}$$

$$0 = \alpha \sum_{l=1}^{2} \sum_{j \in \mathcal{N}_l(i)} \frac{v_j^H - v_i^H}{H_l^2} \; - \; \left(J_{21i}^H \, u_i^H + J_{22i}^H \, v_i^H + f_{2i}^H \right) \tag{22}$$

for $i = 1,...,N^H$, where $((u^H)^\top, (v^H)^\top)^\top = x^H$ and $((f_1^H)^\top, (f_2^H)^\top)^\top = f^H$. The corresponding ordinary and pointwise coupled Gauß-Seidel iteration steps can be derived directly from this system.

(IV) After solving $A^H x^H = f^H$ on the coarse grid, a *prolongation operator* $P^{H \to h}$ has to be defined in order to transfer the solution x^H back to the fine grid:

$$e^h = P^{H \to h} x^H. \tag{23}$$

We choose constant interpolation over $h_x \times h_y$ as prolongation operator $P^{H \to h}$ (see Fig. 1). In particular we have

$$R^{h \to H} = c \cdot (P^{H \to h})^\top \quad \text{with} \quad c = \frac{h_x h_y}{H_x H_y}. \tag{24}$$

The obtained correction e^h can be used now for updating the approximated solution of the original equation on the fine grid:

$$\tilde{x}_{new}^h = \tilde{x}^h + e^h. \tag{25}$$

Finally n_2 iterations of the smoother are performed in order to remove high error frequencies introduced by the prolongation of x^H.

4.3 Strategies for Multigrid Cycles

The hierarchical application of the explained 2-grid cycle is called *V–cycle*. Repeating two 2-grid cycles at *each* level yields the so called *W–cycle*, that has better convergence properties at the expense of slightly increased computational costs (regarding 2D). In general, multiple of such cycles are required to reach the desired accuracy. Examples for V and W–cycles are given in Figure 2.

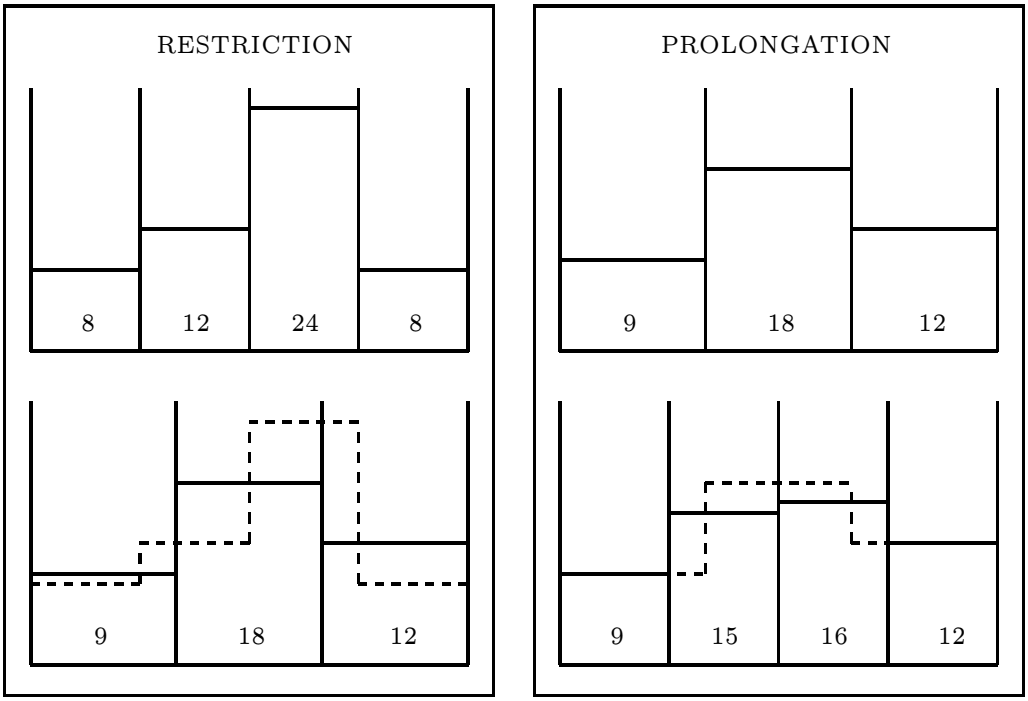

Figure 1: One-dimensional example for the restriction operator $R^{h\to H}$ and the prolongation operator $P^{H\to h}$ with $\frac{h}{H} = \frac{3}{4}$. Numbers represent grey values.

Instead of transferring the residual equations between the levels one may think of starting with a coarse version of the *original* equation system. In this case coarse solutions serve as initial guesses for finer levels. This strategy is referred to as *cascading multigrid*.
In combination with V or W–cycles the multigrid strategy with the best performance is obtained: *full multigrid*. An example for such a full multigrid approach with two W–cycles per level is shown in Figure 3. One should note that the special choice of the restriction and prolongation operator allows a full descent for all three strategies, independently of the problem size. Thus better convergence rates are obtained and the number of required cycles is reduced.

5 Results

We would like to emphasise that the goal of the present paper is to investigate fast algorithms for variational methods. For a detailed evaluation of the CLG

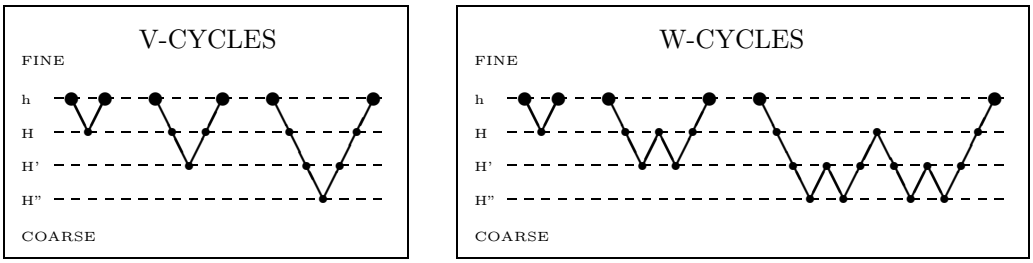

Figure 2: Example of V– and W–cycles for two, three and four levels. In the case of two levels, V– and W–cycles are identical. Performing iterations on the original equation is marked with large black dots, while iterations on residual equations are marked with smaller ones. $H'' > H' > H$ are cell sizes on coarser levels.

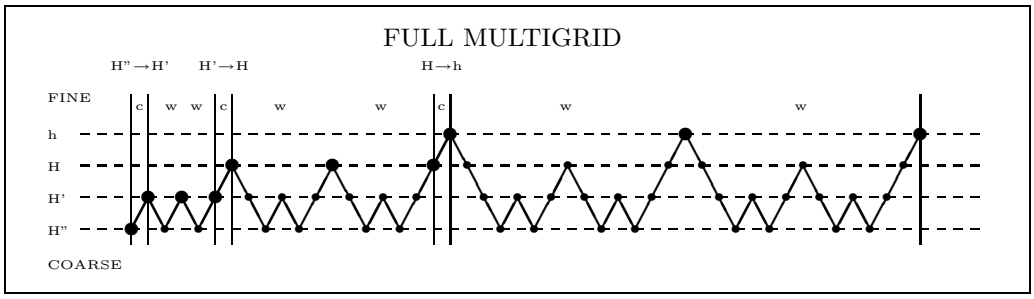

Figure 3: Example of a full multigrid implementation for four levels. Vertical solid lines separate alternating blocks of the two basic strategies. Blocks belonging to the cascading multigrid strategy are marked with c. Starting from a coarse scale the original problem is refined step by step. This is visualised by the $\rightarrow$ symbol. Thereby the coarser solution serves as an initial approximation for the refined problem. At each refinement level two W–cycles are used as solvers (blocks marked with w).

method itself, we refer to [8, 25].
Our computations are performed with C implementations on a standard PC with a 3.06 GHz Intel Pentium 4 CPU. In our first experiment we compare the ordinary and the pointwise coupled Gauß–Seidel method as part of a V-cycle strategy with one presmoothing and one postsmoothing iteration at each level ($n_1 = n_2 = 1$). In order to analyse the convergence behaviour of both numerical schemes with respect to parameter variations, test runs with four different parameter settings have been performed. Each time, the flow

field between frame 16 and frame 17 of the 512×512 *marble* sequence by Otte and Nagel [21] was computed. The V-cycles were stopped when the relative error $e_{rel} := \|e_n\|_2/\|x\|_2$ was below 10^{-3}, where e_n denotes the absolute error after n V-cycles and x stands for the correct solution.

Table 1: Gauß-Seidel vs. pointwise coupled Gauß-Seidel

Parameter			Coupled Gauß–Seidel		Gauß–Seidel	
α	σ	ρ	V-cycles	$\bar{\nu}$	V-cycles	$\bar{\nu}$
1000	2.6	1.8	6	0.152787	6	0.153694
1	2.6	1.8	5	0.110732	30	0.711449
1000	2.6	0.0	6	0.153639	6	0.154601
1	2.6	0.0	23	0.634465	278	0.963437

In Table 1 the required number of V-cycles and the average error reduction factor defined by $\bar{\nu} := \sqrt[n]{\|e_n\|_2/\|e_0\|_2}$ are listed. The superior performance of the pointwise coupled Gauß–Seidel method for decreasing values of α is in accordance with the theoretical considerations from Section 3. In this case, one can also observe an acceleration of the convergence, if the integration scale ρ is used in a moderate way. The reason for this behaviour is the smoothing of the zeroth order term coefficients, which allows a much more accurate coarse grid representation. Thus, the intergation scale is not only useful to render the CLG approach more robust against noise, but also to improve its convergence if multigrid schemes are used.
The computed flow field for the first parameter setting is presented in Figure 4. Since this setting is optimised for quality, its computation time of 0.15 seconds gives already a first impression of the usefulness of multigrid methods for optic flow computation.
In a second experiment different multigrid strategies for the pointwise coupled Gauß–Seidel solver are compared. For this purpose we performed test runs on two different image sequences. Besides the *marble* sequence, the 200×200 pixels *office* sequence by Galvin et al. [12] was used. Table 2 shows the required cycle number n to reach the desired precision of $e_{rel} < 10^{-3}$, the average error reduction factor $\bar{\nu}$ and the run time for each implementation. V/W-cycles are denoted by V/W(n_1,n_2), where n_1 and n_2 stand for the number of pre– respectively postsmoothing iterations. In the case of Full Multigrid strategies an additional number gives information on how many V/W-cycles are used at each refinement level. The presented runtimes refer to the computation of one single flow field of the corresponding sequence.

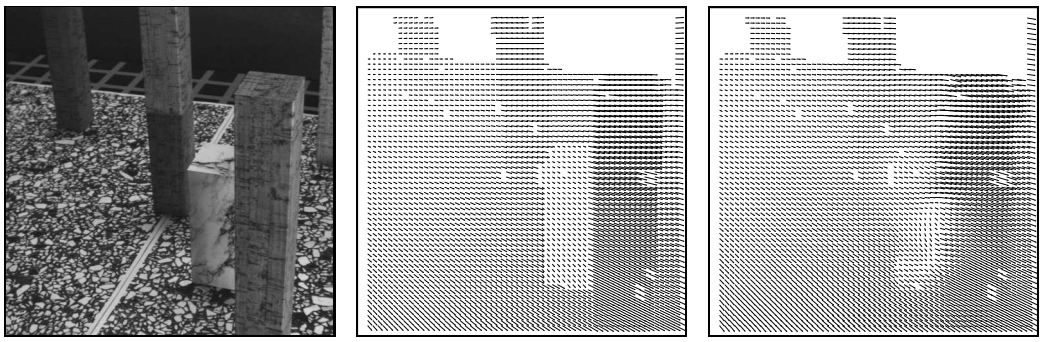

Figure 4: *(a) Left:* Frame 16 of the *marble* sequence. *(b) Center:* Ground truth flow field between frame 16 and 17. *(b) Right:* Computed flow field by our multigrid CLG method ($\sigma = 2.6$, $\rho = 1.8$, and $\alpha = 1000$).

Table 2: Comparison of different multigrid strategies

	Marble			Office		
Strategy	n	$\bar{\nu}$	Time	n	$\bar{\nu}$	Time
V(1,1)	5	0.21946	0.32658	7	0.34182	0.05336
V(2,2)	4	0.11114	0.35697	5	0.20452	0.05561
W(1,1)	1	0.00085	0.16482	2	0.01829	0.03076
W(2,2)	1	0.00067	0.19148	2	0.00877	0.03854
FMG(1,V(2,2))	1	0.00012	0.19947	1	0.00030	0.02436
FMG(2,V(1,1))	1	0.00012	0.22993	1	0.00026	0.02828
FMG(1,W(2,2))	1	0.00012	0.23130	1	0.00025	0.03124
FMG(2,W(1,1))	1	0.00003	0.26929	1	0.00008	0.03788

While strategies based on V-cycles required several iterations to reach a relative error of 10^{-3}, only one or two W-cycles sufficed to perform the same task. This is reflected in significantly shorter run times. The implemented full multigrid schemes converged fastest, but could not outperform the W-Cycles in those cases, where one of them was already sufficient to reach the desired accuracy.

Based on this evaluation we picked the full multigrid implementation with one V(2,2)-cycle each level for our last experiment, where we compared it to the ordinary Gauß–Seidel method and its popular *Successive Overrelaxation* (SOR) variant [28]. Accelerating the Gauß–Seidel method by a weighted extrapolation of its results, the SOR method represents the class of more ad-

vanced non-hierarchical solvers in this comparison.
Table 3 shows the performance of our algorithm, where the presented runtimes refer to all 19 flow fields for the *office* sequence. With more than 40 frames per second we are able to compute the optic flow for sequences with 200×200 pixels in real-time. We see that the chosen multigrid implementation is 247 times faster than the Gauß–Seidel method and almost one order of magnitude more efficient than SOR. In terms of iterations, the difference is even more drastical: While 6692 Gauß–Seidel iterations were required to reach the desired accuracy, a single full multigrid cycle was sufficient.

Table 3: Performance Benchmark

	Iterations/Frame	Runtime $[s]$	Frames/Second $[s^{-1}]$
Gauß–Seidel	6692	115.862	0.164
SOR	152	3.870	4.910
FMG(1,V(2,2))	1	0.470	40.463

Qualitative results for this test run are presented in Figure 5 where one of the computed flow fields is shown. We observe that the CLG method matches the ground truth very well. Thereby one should keep in mind that the full multigrid computation of such a single flow field took only 24 milliseconds.

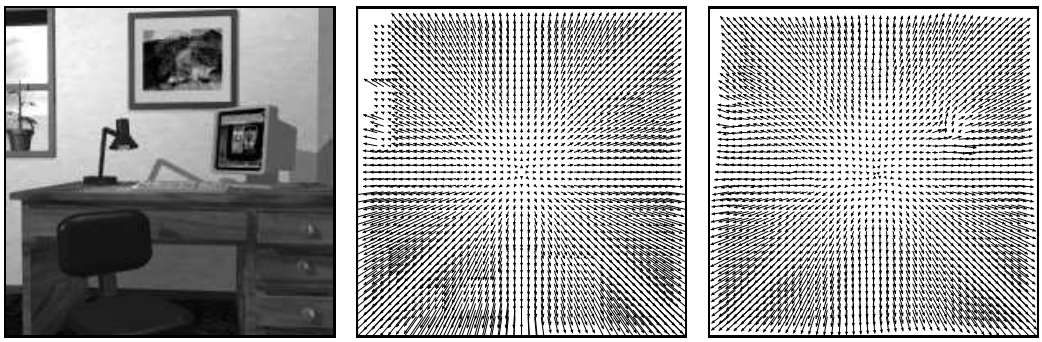

Figure 5: *(a) Left:* Frame 10 of the *office* sequence. *(b) Center:* Ground truth flow field between frame 10 and 11. *(b) Right:* Computed flow field by our full multigrid CLG method ($\sigma = 0.72$, $\rho = 1.8$, and $\alpha = 2700$).

6 Summary and Conclusions

Using the CLG method as a prototype for a noise robust variational technique, we have shown that it is possible to achieve real-time computation of dense optic flow fields of size 200×200 on a standard PC. This has been accomplished by using a full multigrid method for solving the linear systems of equations that result from a discretisation of the Euler–Lagrange equations. We have shown that this gives us a speed-up by more than two orders of magnitude compared to commonly used algorithms for variational optic flow computation. In our future work we plan to investigate further acceleration possibilities by means of suitable parallelisations. Moreover, we will investigate the use of multigrid strategies for nonlinear variational optic flow methods.

Acknowledgements

Our optic flow research is partly funded by the *Deutsche Forschungsgemeinschaft (DFG)* under the project SCHN 457/4-1. This is gratefully acknowledged. Andrés Bruhn thanks Prof. Ulrich Rüde and Dr. Mostafa El Kalmoun from the University of Erlangen for interesting multigrid discussions and Prof. Bernd Finkbeiner from Saarland University for providing the testing environment.

References

[1] P. Ananden. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, Feb. 1994.

[3] J. Bigün, G. H. Granlund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):775–790, Aug. 1991.

[4] A. Borzi, K. Ito, and K. Kunisch. Optimal control formulation for determining optical flow. *SIAM Journal on Scientific Computing*, 24(3):818–847, 2002.

[5] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31(138):333–390, Apr. 1977.

[6] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, Philadelphia, second edition, 2000.

[7] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Real-time optic flow computation with variational methods. In N. Petkov and M. A. Westenberg, editors, *Computer Analysis of Images and Patterns*, volume 2756 of *Lecture Notes in Computer Science*, pages 222–229. Springer, Berlin, 2003.

[8] A. Bruhn, J. Weickert, and C. Schnörr. Combining the advantages of local and global optic flow methods. In L. Van Gool, editor, *Pattern Recognition*, volume 2449 of *Lecture Notes in Computer Science*, pages 454–462. Springer, Berlin, 2002.

[9] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Interscience, New York, 1953.

[10] W. Enkelmann. Investigation of multigrid algorithms for the estimation of optical flow fields in image sequences. *Computer Vision, Graphics and Image Processing*, 43:150–177, 1987.

[11] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS Intercommission Conference on Fast Processing of Photogrammetric Data*, pages 281–305, Interlaken, Switzerland, June 1987.

[12] B. Galvin, B. McCane, K. Novins, D. Mason, and S. Mills. Recovering motion fields: an analysis of eight optical flow algorithms. In *Proc. 1998 British Machine Vision Conference*, Southampton, England, Sept. 1998.

[13] S. Ghosal and P. Č. Vaněk. Scalable algorithm for discontinuous optical flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):181–194, Feb. 1996.

[14] F. Glazer. Multilevel relaxation in low-level computer vision. In A. Rosenfeld, editor, *Multiresolution Image Processing and Analysis*, pages 312–330. Springer, Berlin, 1984.

[15] W. Hackbusch. *Multigrid Methods and Applications*. Springer, New York, 1985.

[16] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[17] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. Seventh International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, Canada, Aug. 1981.

[18] M. R. Luettgen, W. C. Karl, and A. S. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Transactions on Image Processing*, 3(1):41–64, 1994.

[19] E. Mémin and P. Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, May 1998.

[20] H.-H. Nagel. Constraints for the estimation of displacement vector fields from image sequences. In *Proc. Eighth International Joint Conference on Artificial Intelligence*, volume 2, pages 945–951, Karlsruhe, West Germany, August 1983.

[21] M. Otte and H.-H. Nagel. Estimation of optical flow based on higher-order spatiotemporal derivatives in interlaced and non-interlaced image sequences. *Artificial Intelligence*, 78:5–43, 1995.

[22] A. R. Rao and B. G. Schunck. Computing oriented texture fields. *CVGIP: Graphical Models and Image Processing*, 53:157–185, 1991.

[23] D. Terzopoulos. Image analysis using multigrid relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):129–139, Mar. 1986.

[24] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, San Diego, 2001.

[25] J. Weickert, A. Bruhn, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. Technical Report 82, Dept. of Mathematics, Saarland University, Saarbrücken, Germany, Apr. 2003.

[26] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, Dec. 2001.

[27] P. Wesseling. *An Introduction to Multigrid Methods*. Wiley, Chichester, 1992.

[28] D. M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, New York, 1971.

[29] G. Zini, A. Sarti, and C. Lamberti. Application of continuum theory and multi-grid methods to motion evaluation from 3D echocardiography. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 44(2):297–308, Mar. 1997.