FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
INSTITUT FÜR INFORMATIK (MATHEMATISCHE MASCHINEN UND DATENVERARBEITUNG)

Lehrstuhl für Informatik 10 (Systemsimulation)

Parallel multigrid computation of the 3D optical flow

El Mostafa Kalmoun, Harald Köstler and Ulrich Rüde

Lehrstuhlbericht 04-04

Parallel multigrid computation of the 3D optical flow

El Mostafa Kalmoun
Cadi Ayyad University
National School of Applied Science
Marrakech, Morocco

Harald Köstler and Ulrich Rüde
University of Erlangen-Nuremberg
Department of Computer Science 10
Erlangen, Germany

Abstract

This paper deals with parallel 2D and 3D V-cycle multigrid implementation for computing the optical flow between two images. We compare memory costs and convergence rates of four schemes: the Horn-Schunck algorithm (Gauss-Seidel) and multigrid with three different strategies of coarse grid operators discretization: direct coarsening, lumping and Galerkin approaches. Experiments on synthetic images and CT images of the brain are conducted.

Key words. Optical flow, image registration, Horn-Schunck algorithm, multigrid, Galerkin discretization, parallelization.

1 Introduction

Optical flow is commonly defined to be the motion of brightness patterns in a sequence of images. Recent interest has been shown to extend the study of the optical flow from 2D plane images to 3D volume data sets. The main motivation is applications to medical image registration, e.g. for CT images [12], PET volumes [10] and MR images [6].

Differential approaches, which estimate velocity vectors from spatial and temporal intensity derivatives are most used since they allow more efficient exploration of the solution space and result in lower complexity and better accuracy. They are based on brightness conservation assumption which leads to an ill-posed problem that is often solved by imposing additional assumptions that are incorporated into the original problem via regularization techniques. A standard model, which is due to Horn and Schunck [7], results in a system of elliptic PDEs of reaction diffusion type. The second order terms are induced by the regularization and become straightforward Laplace operators. The zero order terms are linear (but variable) and are computed from the derivatives of the image data. Since the image data (and even more so its derivatives) are usually nonsmooth, this poses some nontrivial problems.

The Horn-Schunck algorithm uses a coupled pointwise Gauss-Seidel relaxation to discretize and solve the discretized PDEs. As expected, it has acceptable convergence only if the zero-order terms dominate but the performance is poor, when the diffusion character dominates. In this case, a multigrid strategy can be expected to provide a significant acceleration by allowing a quick propagation of information from non-zero flow field regions into homogeneous or untextured image areas. The Horn-Schunck algorithm can still be used as a smoother for such a multigrid method. The most difficult aspect is to deal with the strongly discontinuous coefficients in an efficient way. In [9], it was shown that using a variational multigrid based on Galerkin approach leads to a good and stable performance on all sort of images. However, to the best of our knowledge, an investigation of multigrid computation of the optical flow on parallel machines is still-lacking. This paper reports experimental progress in implementing different multigrid solvers for the Horn-Schunck model. We have considered this model for its simplicity and popularity.

The outline of the paper is as follows. First, we present in Section 2 an extension of the Horn-Schunck model to the 3D case. In Section 3, we propose a variational multigrid scheme for this model based on the Galerkin approach. Then, we describe in Section 4 parallelization of this scheme and give some implementation details. In Section 5, we present comparison results from some experiments conducted on synthetic and medical data sets. Finally, we conclude the paper in Section 6 and give an outlook of future work.

2 The 3D Horn-Schunck model

We are extending here the well-known optical flow constraint equation to the three-dimensional case. Assume we are given a sequence of 3D images with an intensity value $I(x, y, z, t)$ of the volumetric image point (x, y) at time t. The intensity values $I(x, y, z, t)$ are supposed to be described by a differentiable function $I : \Omega \times [0, T] \rightarrow \mathbf{R}$, where $\Omega \subset \mathbf{R}^3$ is the imaging volume and T is a strictly positive scalar describing the final time. The partial derivatives of I in the direction of x, y, z and t are denoted by I_x, I_y, I_z and I_t, respectively. The underlying optical flow assumption says that image objects keep the same intensity value under motion for at least a short period of time. In terms of equations, this can be stated as follows: $\forall (x, y, z) \in \Omega, \forall t \in [0, T]$,

$$I(x, y, z, t) = I(x + dx, y + dy, z + dz, t + dt).$$

Using Taylor expansion and dropping the nonlinear terms, we get the 3D optical flow constraint equation (OFCE):

$$I_x u + I_y v + I_z w + I_t = 0,$$

where $(u, v, w) = (\frac{dx}{dt}, \frac{dy}{dt}, \frac{dz}{dt})$ is the 3D velocity vector. This equation defines an ill-posed problem which is solved via regularization [14]. The optical flow is supposed to have smooth variations in the sense that neighboring points have almost the same velocity. The (OFCE) is hence replaced by the following minimization problem:

$$\min_{(u,v,w)} \int_x \int_y \int_z \left[(I_x u + I_y v + I_w z + I_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2 + |\nabla w|^2) \right] \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}z, \tag{1}$$

where α is a positive parameter for adjustment between the data and the additional smoothness constraint. In general, it would be required to interactively adjust α to find the best value.

Remark 2.1. In the context of image registration, when displacements between the reference and template images are expected to be very large, one has to consider the data term to be the squared difference of the two images without using the 1st order Taylor expansion. The corresponding minimization problem will be then as follows:

$$\min_{(u,v,w)} \int_x \int_y \int_z \left[(I(x, y, z, t) - I(x + u, y + v, z + w, t + 1))^2 + \alpha(|\nabla u|^2 + |\nabla v|^2 + |\nabla w|^2) \right] \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}z,$$

Euler-Lagrange equations derived from the minimization problem (1) define a system of three elliptic PDEs with nonconstant coefficients depending on the volumetric image data for the zero-order terms:

$$\begin{aligned} \alpha \Delta u - I_x(I_x u + I_y v + I_z w + I_t) &= 0 \\ \alpha \Delta v - I_y(I_x u + I_y v + I_z w + I_t) &= 0 \\ \alpha \Delta w - I_z(I_x u + I_y v + I_z w + I_t) &= 0. \end{aligned} \tag{2}$$

This system is symmetric with respect to the three components of the velocity u, v and w. Thus, a standard way to solve it is the block Gauss-Seidel relaxation:

$$\begin{aligned} u^{k+1} &= \bar{u}^k - I_x \, \frac{I_x \bar{u}^k + I_y \bar{v}^k + I_z \bar{w}^k + I_t}{\alpha + I_x^2 + I_y^2 + I_z^2} \\ v^{k+1} &= \bar{v}^k - I_y \, \frac{I_x \bar{u}^k + I_y \bar{v}^k + I_z \bar{w}^k + I_t}{\alpha + I_x^2 + I_y^2 + I_z^2} \\ w^{k+1} &= \bar{w}^k - I_z \, \frac{I_x \bar{u}^k + I_y \bar{v}^k + I_z \bar{w}^k + I_t}{\alpha + I_x^2 + I_y^2 + I_z^2}, \end{aligned}$$

where $\bar{u}$ (resp. $\bar{v}, \bar{w}$) denotes an average of the neighboring points to u (resp. v, w).
This relaxation scheme is known to have slow convergence if the diffusion term dominates. We propose then to use a multigrid method to speed up the convergence.

3 A variational multigrid scheme

Multigrid methods are known to be among the most efficient solution methods for elliptic PDEs. For a comprehensive overview on multigrid methods, we refer to [2] and [15]. The core idea of multigrid is to use a sequence of coarse grids as a means to accelerate the solution process (by relaxation such as Gauss-Seidel) on the finest grid. This leads to recursive algorithms like the so-called V- or W-cycle, which traverse between fine and coarse grids in the mesh hierarchy. Since ultimately only a small number of relaxation steps on each level must be performed, multigrid provides an asymptotically optimal method whose complexity is only $O(N)$, where N is the number of mesh points.

The first attempts to use multilevel techniques for low-level problems in computer vision, and particularly for the optical flow are due to Glazer [5] and Terzopoulos [13]. They both reported improvement in performance but only by testing simple synthetic images. Enkelmann [4] adopted a coarse-to-fine strategy and used an image pyramid to subsample the images into different resolutions. The spatial intensity gradient information need not to be transferred to the current coarse level as in [5] since they could be calculated from the corresponding level of the image pyramid. Experiences on real 2D images have shown that both methods (direct restriction of the image gradient and building an image pyramid) when used in a standard multigrid lead to similar results: acceptable convergence rate for *smooth* images and slow convergence or even divergence for highly-textured images, especially if we consider more than two levels in the multigrid hierarchy. Clearly, there is a loss of information when we represent the problem on coarse levels. Battiti *et al* [1] stated that a usual multigrid cycle is not appropriate due to a possible information conflict between different scales, and they preferred rather a one-way (coarse-to-fine) strategy. However, one-way multigrid methods - also known as cascadic multigrid, do not reach the optimal efficiency of a classical (bidirectional) multigrid method with a good coarse grid approximation of the original problem. A close look on system (2) shows that the finest grid operator has some nice properties that could be exploited for coarse grid approximation; we refer to [9] for the 2D case and develop that here for the 3D case.

First, we write system (2) as follows

$$L\xi = F \tag{3}$$

where

$$\xi = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \quad , \quad F = \begin{pmatrix} -I_x I_t \\ -I_y I_t \\ -I_y I_z \end{pmatrix},$$

$$L = L_d + L_r \quad , \quad L_r = \begin{pmatrix} -\alpha\Delta & 0 & 0 \\ 0 & -\alpha\Delta & 0 \\ 0 & 0 & -\alpha\Delta \end{pmatrix},$$

and L_d is a 3x3 block diagonal matrix with entries

$$\begin{pmatrix} I_x^2 & I_x I_y & I_x I_z \\ I_x I_y & I_y^2 & I_y I_z \\ I_x I_z & I_y I_z & I_z^2 \end{pmatrix}$$

It is clear that the term L_r is symmetric positive definite and one can show that L_d is symmetric positive semi-definite. Hence the sum L is also symmetric positive definite. This suggests to put the linear system (3) into its equivalent variational minimization form

$$\xi = arg \min_{\Omega} a(\eta).$$

Here a is the quadratic form given by $a(\eta) = \frac{1}{2}(L\eta, \eta) - (F, \eta)$.
Let's denote by h and H the current resolution and the next coarser resolution, respectively. Let also $I_H^h : \Omega^H \mapsto \Omega^h$ be a full rank linear mapping. An optimal coarse grid correction $I_H^h \xi_H$ of the current approximation ξ_h is characterized by

$$((I_H^h)^T L_h I_H^h)\xi_H = (I_H^h)^T \left(F - L_h \xi_h\right).$$

The Galerkin approach consists then to choose the coarse grid operator as follows

$$L_H = I_h^H L_h I_H^h \quad and \quad I_h^H = (I_H^h)^T.$$

For our system, this yields

$$\begin{aligned} L_H &= \begin{pmatrix} R & 0 & 0 \\ 0 & R & 0 \\ 0 & 0 & R \end{pmatrix} \begin{pmatrix} L_h^1 & L_h^2 & L_h^3 \\ L_h^2 & L_h^4 & L_h^5 \\ L_h^3 & L_h^5 & L_h^6 \end{pmatrix} \begin{pmatrix} P & 0 & 0 \\ 0 & P & 0 \\ 0 & 0 & P \end{pmatrix} \\ &= \begin{pmatrix} R L_h^1 P & R L_h^2 P & R L_h^3 P \\ R L_h^2 P & R L_h^4 P & R L_h^5 P \\ R L_h^3 P & R L_h^5 P & R L_h^6 P \end{pmatrix} \end{aligned}$$

where R and P are any 3D-grid transfer operators satisfying $P = R^T$. In our implementation, R is the full weighting and P is the bilinear interpolation, see [15]. The components of our V-cycle are described as follows

- Vertex-centered grid and standard coarsening
- Coupled lexicographic point Gauss-Seidel smoother
- Full weighting and bilinear interpolation
- Galerkin coarse grid approximation (GCA approach)

4 Implementation issues and parallelization

We discretize the Laplacian Δ by the standard 7 point stencil, which means that the averages $\bar{u}$, $\bar{v}$ and $\bar{w}$ are computed from the stencil

$$\frac{1}{6}\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right). \tag{4}$$

As suggested by Horn and Schunck, we calculate spatial and temporal image derivatives halfway between pixels in the x, y, z, t directions. In our implementation, all images are presmoothed with a Gaussian Kernel before applying convolution masks to compute the derivatives. Precisely, we use the following masks

$$M_x = \frac{1}{8}\left(\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \right) \quad M_y = \frac{1}{8}\left(\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \right)$$

$$M_z = \frac{1}{8}\left(\begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \right) \quad M_t = \frac{1}{8}\left(\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \right)$$

and then the partial derivatives are calculated by

$$I_x = M_x * (I_1 + I_2) \quad I_y = M_y * (I_1 + I_2) \quad I_y = M_z * (I_1 + I_2) \quad I_t = M_t * (I_2 - I_1).$$

The code is written in C++ language and parallelized with MPI ([17],[18]). We adopt for the parallel implementation of the multigrid V-cycle a grid partitioning strategy rather than domain decomposition because it is known to keep the asymptotic convergence behavior of the sequential algorithm [11]. The image domain is split up into parts along all directions and a layer of ghost cells on each boundary between two processors is introduced to store the outermost values of the neighboring processor. For the parallel smoother, we use a red-black (in 3D a multicolor) Gauss-Seidel solver instead of a lexicographic one because of the dependencies of the unknowns. After the update of each color in a Gauss-Seidel step, the corresponding ghost cells are exchanged between neighboring processors. In addition to this data exchange in the smoother we have to exchange on each level in the V-cycle the computed residual and the corrected solution (see Algorithm 1). Especially for a bigger number of coarse levels the amount of communication needed for parallel multigrid gets very high. A detailed discussion of the parallelization of multigrid is e.g. found in [15].

The full scheme is described in the following.

Algorithm 1 Parallel V-cycle

```
1: Presmoothing {One exchanges the ghost cells of each color}
2: Compute residual
3: Restrict residual
4: Exchange residual
5:
6: if on coarsest level then
7:    smooth several times {One exchanges the ghost cells of each color}
8: else
9:    call V-cycle recursively on next coarser level
10: end if
11:
12: Correct current solution with interpolated solution from coarser level
13: Exchange solution
14: Postsmoothing {One exchanges the ghost cells of each color}
```

1. Main processor reads in frames
2. Mesh partitioning and the main processor distributes the needed parts of the frames to the other processors
3. Each processor computes spatial and temporal derivatives
4. Each processor calculates the finest grid operator and the right hand side (RHS)
5. Each processor sets up the coarse grid operators hierarchy using the Galerkin scheme
6. Each processor participates in the multigrid V-cycling
7. The results of each processor are collected by the main processor
8. Main processor writes out the optical flow field

A challenging problem when using standard multigrid to solve the optical flow problem is the bad convergence rate due to the possible non-smoothness of the spatial and temporal image derivatives. To overcome this problem we have introduced in Section 3 the Galerkin coarse grid approximation. The disadvantage of this approach is the growing memory cost as shown in the following comparison of memory used for a number of different methods.

Let $d \in \{2, 3\}$ be the dimension of the data set and let N be the number of image points. S denotes the number of bytes needed to store one image point.
For the Horn and Schunck method, we need $d + 1$ grids for the image derivatives and d grids for the solution. This leads to a total memory cost of

$$M_H = S \cdot N(d + d + 1) \ . \tag{5}$$

For multigrid with l levels, we need d grids for the solution and additionally d grids for the RHS on each level. Our first two approaches implemented so far to reduce memory cost are lumping and straightforward direct coarsening. For direct coarsening we simply restrict the image derivatives to coarse grid and use a constant $2d+1$ point stencil. The I_t derivative is not longer needed, because that information is included in the RHS. Thus we have

$$M_D = S \cdot N \left(d + 2d \left(\sum_{i=0}^{l} \frac{1}{2^{id}} \right) \right) \ , \tag{6}$$

but the convergence rate gets much worse, since the coarse grid operator does not approximate the fine grid operator in an appropriate way.

sizes 1000x1000 and 100x100x100				
Method	Memory (MB)	Factor 2D	Memory (MB)	Factor 3D
Horn-Schunck	38.15	5	53.41	7
Galerkin	123.39	16.17	251.43	32.95
Lumping	63.31	8.30	81.72	10.71
Direct	55.79	7.31	75.20	9.86

Table 1: Comparison of memory cost.

$\alpha = 1$ and $u_0 \neq v_0 \neq w_0$		
Method	Convergence rate 2D	Convergence rate 3D
Horn-Schunck	0.996	0.998
Galerkin	0.059	0.12
Lumping	0.096	0.158
Direct	0.096	0.158

Table 2: Comparison of convergence rates in 2D and 3D for the test problem $I_x = I_y = I_z = I_t = 1$.

Lumping that is a well-known technique in the context of finite elements (cf. [3], [8]) reduces the size of the stencils obtained with Galerkin coarsening of L_d by summing up the off-center entries and add it to the center value. For L_r direct coarsening is used. Here we get

$$M_L = S \cdot N \left(d + 2d \left(\sum_{i=0}^{l} \frac{1}{2^{id}} \right) + \frac{d^2 + d}{2} \left(\sum_{i=1}^{l} \frac{1}{2^{id}} \right) \right) . \tag{7}$$

When using the full Galerkin coarsening, a 3^d point stencil has to be stored for each grid point except on the finest grid

$$M_G = S \cdot N \left(d + 2d \left(\sum_{i=0}^{l} \frac{1}{2^{id}} \right) + \frac{d^2 + d}{2} \cdot 3^d \left(\sum_{i=1}^{l} \frac{1}{2^{id}} \right) \right) . \tag{8}$$

Table 1 presents a comparison of the memory cost of the methods for image sizes 1000x1000 in 2D and 100x100x100 in 3D using double precision (S = 8 Bytes). Thus we need about 7.63 MB to store a single grid. For all multigrids we used 4 levels ($l = 4$). The given factors are the amount of storage needed divided by the memory cost of one single grid.

Table 2 shows the convergence rates of the different methods for the test problem $I_x = I_y = I_z = I_t = 1$ of size 65x65 in 2D and 65x65x65 in 3D. We use here a $V(2, 1)$ V-cycle with 5 levels.

5 Experimental results

All parallel experiments were performed on a SGI Altix 3700 supercluster that consists of 7 nodes and 4 CPUs on each. The nodes are linked together via a high speed network of type NUMALink3. The processors are of type Itanium2 “Madison” with 1.3 GHz clock rate and 5.2 GFlop/s peak performance. The machine has 112 GByte ”distributed shared” memory. For more details we refer to [16].

For characterizing the capability of our parallelization we use the speedup

$$s = \frac{t_1}{t_p} ,$$

where t_1 is the time needed for the algorithm on one processor and t_p is the time for p processors. A measure for the workload on each processor is the efficiency

$$e = \frac{s}{p} .$$

We tested our parallel multigrid implementation using Galerkin discretization on two volume data sets with different sizes. Th first one is the 100x100x100 test problem for which we have $I_x = I_y = I_z = I_t = 1$. The second volumes are two computed tomography (CT) data sets used for brain angiography studies with a size equal to 512x512x88 (Figure 1). We have chosen the regularization scalar $\alpha = 4$.
In Table 3 the mean time in seconds for a single V-cycle with 4 levels is presented. From 1 to 2 processors we see a good speedup since we use a lot of memory per CPU here. Starting with 8 processors the communication overhead begins to show in the timings. In comparison to that one Horn and Schunck Gauss Seidel step for the medical data takes about 11 seconds on a single processor.

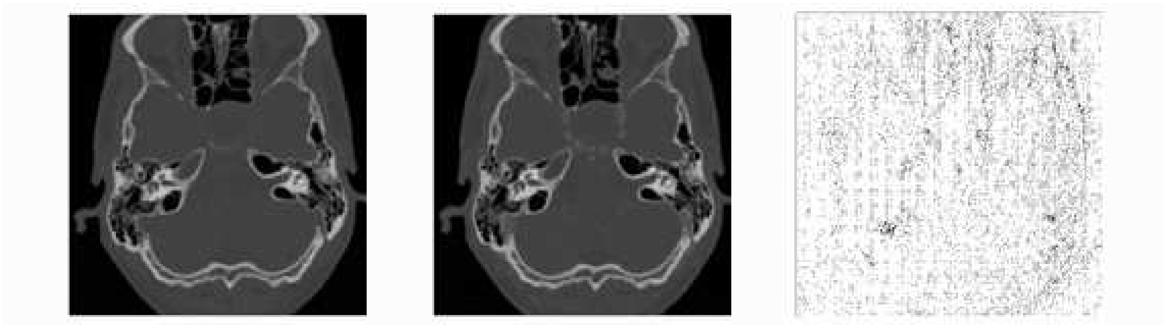

Figure 1: CT data sets used for brain angiography studies. Left: One slice of the first volume where the scan is done with contrast agent. Middle: Corresponding slice in the second volume where the scan is without contrast agent. Right: The corresponding slice of the 3D optical flow.

	Size 100x100x100			Size 512x512x88		
No. CPUs	Time (sec.)	Speedup s	Efficiency e	Time (sec.)	Speedup s	Efficiency e
1	5.7			265		
2	3.9	1.46	0.73	133	1.99	0.99
4	2.0	2.85	0.71	82	3.23	0.81
8	1.1	5.18	0.65	65	4.08	0.51
16	0.7	8.14	0.51	44	6.02	0.38

Table 3: Time in seconds, Speedup and Efficiency on Altix for different problem sizes in 3D using a V(2,2) cycle.

6 Conclusion

We have reported our experience with the implementation of the 2D and 3D optical flow on parallel machines and have shown that optical flow techniques can be used for registration of real medical data sets. We have also compared memory costs and convergence rates of four schemes: the Horn-Schunck algorithm (Gauss-Seidel) and multigrid with three different strategies of coarse grid operators discretization: direct coarsening, lumping and Galerkin approaches.

Experimental results for both 2D and 3D data sets have shown encouraging results when using Galerkin discretization with respect to the asymptotic behavior. The convergence rate is relatively good although one can slightly improve it by implementing operator dependent restriction and interpolation operators. However, due to high memory cost, the overall performance of the (parallel) variational multigrid method was negatively affected in the 3D case and for large volume data sets. A compromise between a good representation of the finest grid operator on coarser levels and an optimal memory storage should be investigated. This will be addressed in the future work.

Acknowledgments

The authors wish to thank Dr. Yassine Bouchareb from the Institute of Medical Physics at Erlangen University for providing the CT data sets of the brain. Thanks are due also to Michael Bauer and Frank Firsching from Department of Computer Graphics for helping with visualization of 3D data. Finally, the first author gratefully acknowledges the financial support from the Graduate Research Center "3D Image Analysis and Synthesis" during his stay at Erlangen University in August 2004.

References

[1] Battiti R, Amaldi E, Koch C. Computing optical flow across multiple scales: an adaptive coarse-to-fine strategy, *International Journal of Computer Vision* 1991, **6**:133-145.

[2] Briggs W.L, Hensen V.E, McCormick S.F. *A multigrid tutorial* . SIAM, 2000.

[3] Ciarlet P.G. *The finite element method for elliptic problems.* North Holland, Amsterdam, 1978.

[4] Enkelmann W. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences, *Computer Vision, Graphics, and Image Processing* 1988, **43**:150-177.

[5] Glazer F. Multilevel relaxation in low-level computer vision, in *Multi-resolution image processing and Analysis (A. Rosenfeld, Ed.) Springer-Verlag* 1984, 312-330.

[6] Hata N, Nabavi A, Wells W.W, Warfield S, Kikinis R, Black P, Jolesz F.A. Three-dimensional optical flow method for measurement of volumetric brain deformation from intraoperative magnetic resonance images, *J Comput Assist Tomogr.* 2000, **24**:531-538.

[7] Horn B.K.P, Schunck B.G. Determining optical flow, *Artificial Intelligence* 1981, **17**:185-203.

[8] Hughes T. *The finite element method: linear static and dynamic finite element analysis.* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[9] Kalmoun E.M, Rüde U. A variational multigrid for computing the optical flow, In: *Vision, Modeling and Visualization 2003.* T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, R. Westermann (Eds.). Akademische Verlagsgesellschaft, Berlin 2003, p.p 577-584.

[10] Klein G.J, Huesman R.H. A 3D optical flow approach to addition of deformable PET volumes, *Proceedings of the 1997 IEEE Workshop on Motion of Non-Rigid and Articulated Objects (NAM '97)*, June 1997.

[11] Liorante I.M, Tirado F. Relationships between efficiency and execution time of full multigrid methods on parallel computers, *IEEE Trans. on Parallel and Distributed Systems* 1997,**8**:562-573.

[12] Song S.M, Leahy R.M. Computation of 3-D velocity fields from 3-D cine CT images of the human heart, *IEEE Transactions on Medical Imaging* 1991,**10**(3):295-306.

[13] Terzopoulos D. Image analysis using multigrid methods, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1986, **8**:129-139.

[14] Tikhonov A.N, Arsenin V.Y. *Solution of ill-posed problems*, Wiley:New York, 1977.

[15] Trottenberg U, Oosterelee C, Schüller A. *Multigrid.* Academic Press, 2001.

[16] *http://www.rrze.uni-erlangen.de/dienste/arbeiten-rechnen/hpc/systeme/sgi-altix-3700.shtml*

[17] The MPI Forum. The MPI message-passing interface standard. *http://www.mcs.anl.gov/mpi/standard.html*, May 1995

[18] Gropp W. *Using MPI-2.* MIT Press, 1999.