



TMA4205 — Fall 2017

## STUDENT PROJECT

Group members:

★ 750883

★ 486343

### Abstract

Optical flow describes apparent motion in an image and can be used for object segmentation and collision detection. The project compares multigrid and conjugate gradient methods for a numerical estimation of the optical flow with respect to their efficiency. We analyze convergence and calculation time for implementations of those methods on artificial testimages and one example from a real film. **The preconditioned conjugate gradient method is the most accurate and a pure V-cycle is the fastest implementation.**

### INTRODUCTION

Projecting a three dimensional scene with movement to an image plane with relative movement, for example with a camera or an eye results in an apparent motion within the image plane called optical flow. The optical flow equation cannot be expected to have an unique solution, since information is essentially just given across the edges and not along them. This property of the equation is called aperture problem. We therefore interpret the estimation of optical flow as an minimization problem. We analyze the optical flow for greyscale images and due to the structure of the image data, we use finite differences to approximate the equation and employ multigrid and conjugate gradient methods for an accurate, more importantly efficient solution. We create gaussian test images and analyze convergence and calculation time of those methods and verify the results with reallife data from a movie.

### THEORY

#### General theory

Denote by  $\Omega := [0, L_x] \times [0, L_y] \subset \mathbb{R}^2$  the plane of an image, and assume we are given a sequence of images on  $\Omega$  such that  $\Omega \times \mathbb{R} \rightarrow \mathbb{R}$ . At some fixed time, the vector field that is the optical flow, may be written as  $\omega$ .  $\omega$  has components  $u$  and  $v$  such that

$$\omega = \begin{bmatrix} u(x, y) \\ v(x, y) \end{bmatrix}.$$

With  $u$  and  $v$  namely the horizontal and the vertical apparent movement of each pixel in the movie.

If we assume the intensity of the picture is closed to constant at is, we assume the time step to be small, we may write  $I$  as

$$I(x, y, t_0) = I(x + \Delta u(x, y), y + \Delta v(x, y), t_0 + \Delta t_0).$$

If we divide by  $\Delta t_0$  and let  $\Delta t_0 \rightarrow 0$ , we get

$$u \frac{\partial I}{\partial x} + v \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0.$$

This is however only one equation, but two unknown functions. To be able to define  $u$  and  $v$  we need to add an extra constraint. A solution would be to add smoothness to the solution, meaning that the values of  $u$  and  $v$  should only change slowly in space. A possibility is therefore to demand that  $u$  and  $v$  solve the optimization problem

$$\frac{1}{2} \|u \partial_x I + v \partial_y I + \partial_t I\| + \frac{\lambda}{2} (\|\nabla u\|^2 + \|\nabla v\|^2) \rightarrow \min.$$

This leaves us with the coupled PDEs

$$\begin{aligned} (u \partial_x I + v \partial_y I) \partial_x I - \lambda \Delta u &= -\partial_x I \partial_t I \\ (u \partial_x I + v \partial_y I) \partial_y I - \lambda \Delta v &= -\partial_y I \partial_t I. \end{aligned}$$

in this paper, only homogeneous Dirichlet boundary conditions will be considered. With homogeneous Dirichlet boundary conditions we assume there are no flow at the boundary of the image.

## Discretisation

For the discretisation we assume we are only given two consecutive frames, namely  $I_0$  and  $I_1$ , of a movie at times  $t = 0$  and  $t = \Delta t$ . The images are greyscale and we will assume the pictures are rectangular with homogeneous length between the center of each pixel. For simplicity we will assume that this distance is 1. We also assume  $\Delta t = 1$ . If we approximate the partial derivatives of  $I$  with finite differences we get for the  $t$  derivative

$$\partial_t I(x_i, y_i) = I_1(x_i, y_i) - I_0(x_i, y_i)$$

For the  $x$  derivative we get

$$\partial_x I_0(x_i, y_i) = \begin{cases} I_0(x_{i+1}, y_i) - I_0(x_i, y_i) & \text{if } i < m \\ I_0(x_m, y_i) - I_0(x_{m-1}, y_i) & \text{if } i = m \end{cases}$$

and

$$\partial_x I_1(x_i, y_i) = \begin{cases} I_1(x_{i+1}, y_i) - I_1(x_i, y_i) & \text{if } i < m \\ I_1(x_m, y_i) - I_1(x_{m-1}, y_i) & \text{if } i = m \end{cases}$$

We will use the mean of this as  $I_x$ . The same procedure can be done for  $I_y$ .

To discretize the Laplace operator, the standard five point stencil is used.

We are therefore left with the coupled system

$$\begin{aligned} (\partial_x I)_{ij}^2 u_{ij} + (\partial_x I)_{ij} (\partial_y I)_{ij} v_{ij} - \lambda (A_h u)_{ij} &= -(\partial_t I)_{ij} (\partial_x I)_{ij} \\ (\partial_x I)_{ij} (\partial_y I)_{ij} u_{ij} + (\partial_y I)_{ij}^2 v_{ij} - \lambda (A_h v)_{ij} &= -(\partial_t I)_{ij} (\partial_y I)_{ij} \end{aligned} \quad (1)$$

### Properties of the system

We are able to write (1) in matrix form as

$$\begin{bmatrix} (\partial_x I)_{ij}^2 & (\partial_x I)_{ij} (\partial_y I)_{ij} \\ (\partial_x I)_{ij} (\partial_y I)_{ij} & (\partial_y I)_{ij}^2 \end{bmatrix} \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix} - \lambda \begin{bmatrix} (A_h u)_{ij} \\ (A_h v)_{ij} \end{bmatrix} = - \begin{bmatrix} (\partial_t I)_{ij} (\partial_x I)_{ij} \\ (\partial_t I)_{ij} (\partial_y I)_{ij} \end{bmatrix} \quad (2)$$

It is well known that the discretisation of the Laplacian is symmetric. It is also visible that the matrix in the first term of (2) is symmetric. It is therefore possible to conclude that the system is symmetric.

As for positive definiteness, it is well known that the negative discretization of the laplacian is positive definite, and therefore it is symmetric positive definite (SPD) for the matrix coming from the first term, we see that the eigenvalues are

$$\lambda_1 = (\partial_x I)_{ij}^2 + (\partial_y I)_{ij}^2 \quad \text{and} \quad \lambda_2 = 0.$$

Since the matrix is symmetric we are only dealing with real eigenvalues values,  $\lambda_1$  is therefore greater than zero. We conclude that  $B_{ij}$ , defined as the first term in (2), is symmetric positive semi-definite (SPSD).

If we now reshape  $u$  and  $v$  to be column vectors then we may write the full system as,

$$B \begin{bmatrix} u \\ v \end{bmatrix} - \lambda A_h \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t I_x \\ I_t I_y \end{bmatrix} \quad (3)$$

The values in each  $B_{ij}$  is placed in the rows and columns corresponding to their respective  $u_{ij}$  and  $v_{ij}$  in a sparse matrix  $\hat{B}_{ij}$ .  $B$  is therefore defined as the sum of all sparse matrices  $\hat{B}_{ij}$  which all are SPSPD. It is then trivial to see that the matrix  $B$  also has this property. Since  $A_h$  is SPD we see that the total system is SPD. Numerical solvers like conjugate gradient method will therefore converge for (3).

For diagonal dominance we assume it is known that the  $A_h$  is irreducibly row diagonally dominant. As for the  $B$  matrix which is a block matrix of the form

$$B = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

where each block is a diagonal matrix. We see that there are three cases either  $I_{yi} > I_{xi}$  or  $I_{yi} < I_{xi}$  or  $I_{yi} = I_{xi}$  for some row  $i$ . For the first case the upper part of  $B$  won't be diagonally dominant and for the second case the lower part won't be diagonally dominant. If we are in the last case for all  $i$  the matrix will be neither diagonally dominant nor irreducibly row diagonally dominant. The whole system will therefore never be diagonally dominant, and only irreducibly row diagonally dominant if

all  $I_{xi} = I_{yi}$ .

For an iterative method like Gauss-Seidel or Jacobi we can guarantee convergence of the system if the matrix is either irreducibly diagonally dominant, diagonally dominant or SPD. Though possible, the system will probably never be irreducibly diagonal dominant, but the convergence is saved by the symmetric positive definiteness of the system.



## Iterative solvers

We can solve the System using the conjugate gradient method, because our System is positive definite and symmetric. These properties guarantee us convergence, but the implementation is not optimal due to the nonlinear increase in computation time for large systems. Especially for pictures of high resolution we can expect to run into difficulties analyzing a whole film sequence.

A second way to solve this system is to use a multigrid method, where we use different mesh sizes to obtain optimal convergence in contrast to a pure preconditioned Krylov method, which slows down significantly for large grid sizes. We therefore use different discretizations of the original problem and translate the solution between the grids with restriction and prolongation operators. Low frequency modes of the error or residual are naturally mapped to high frequency modes as we restrict our System to the coarser grid.

If we then use iterative methods we are able to smooth the high frequency modes on the coarser grid and then can prolong the results to the finer grids. These iterative methods will be referred to as "smoothers". The Translation process between the different discretizations introduces errors which we try to resolve with smoothing algorithms. For our problem we start with the discretization described previously as the finest grid. It doesn't seem reasonable to start with a finer grid, since the information that can be obtained from the images is discrete and corresponds directly to our nodes in the mesh currently described. The size of the image data corresponds to power of two, this wouldn't be the case we could add a padding on the outside of the image to receive edge lengths divisible by two. We derive the restriction operator for our finite difference discretization from the stencil


$$\frac{1}{4} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad (4)$$




and receive a block diagonal matrix with blocks



$$B = \frac{1}{4} \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & & \\ & \ddots & \ddots & & \ddots & \ddots & \\ & & 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \quad (5)$$

with length  $2m$  and height  $n$  for an image of the size  $m$



$$\begin{bmatrix} [B] & [B] & & \\ & \ddots & & \\ & & [B] & [B] \end{bmatrix}. \quad (6)$$

The prolongation operator equals four times the transpose of the restriction operator. We use a V-cycle where we step down  the residual equation to the coarsest level using a smoother at each level. We then solve the system exactly at the coarsest level with the conjugate gradient method, and translate the updates to the finest grid using the same smoother as we move up.

The  system properties are preserved under our restriction operator which guarantees us convergence of the method, since we have convergence for the basic iterative method and the Conjugate Gradient method. The amount of steps used for pre- and post-  smoothing as well as the amount of levels used for the V-cycle will be analyzed heuristically with respect to the actual image data for our problem. 

A good choice for a smoother will in this case be the Red-Black Gauß-Seidel method. This method is based on coloring each node on the grid to a given color in a way that no neighbouring nodes have  the same color. Though possible to only use two colors, there will be used four different colors in the method described here to ease the implementation task. If none of the equally colored nodes neighbour each other, then we may compute the value for all the nodes of one color in parallel. This method will therefore improve the speed of the standard Gauss-Seidel method and still keep the same convergence criteria. The convergence is also expected to be faster since the newest update is always  being used in the iterations.

From the theory of the Gauss-Seidel method we know that the first few iterations smooths out the high frequency oscillations of the solution. The method does however use a lot more time smoothing out the lower oscillations. In the multigrid method we will therefor only use a few iterations of the smoother for each level.

A third way to get a solution would be the PCG method, where we use the V-cycle as a  precondition for the conjugate gradient method. We solve the actual equation at the coarsest grid instead of computing updates with the residual equation  with this procedure we can damp the low frequency modes of the error much better. With the approach we accelerate the convergence and can expect this algorithm to outperform the pure V-cycle. Convergence properties again follow from the results about RB-Gauß-Seidel and the conjugate gradient algorithm. We employ the generic algorithm for the PCG method but compute

$$z \leftarrow M^{-1}r \quad (7)$$

with one iteration of the V-cycle with   $M$  being the actual System Matrix.

---

## IMPLEMENTATION AND NUMERICAL EXPERIMENTS

---

First we analyze the general accuracy and the convergence speed for the presented methods for a real life example. In contrast to the pure conjugate gradient algorithm we can see the almost linear convergence for V-cycle and PCG.

V-cycle preconditioner change V-cycle post conditioner change V-cycle conditioner change V-cycle level change V-cycle level and preconditioner change

---

## CONCLUSION

---

---

**REFERENCES**

---

- [1] *Eigenvalues of tridiagonal Toeplitz matrices* retrieved from <https://www.math.ntnu.no/emner/TMA4205/2015h/notes/tridiag.pdf>.