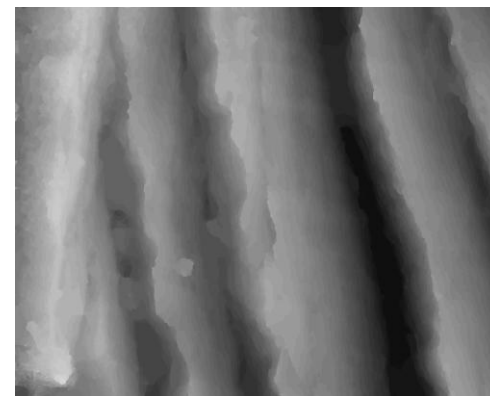
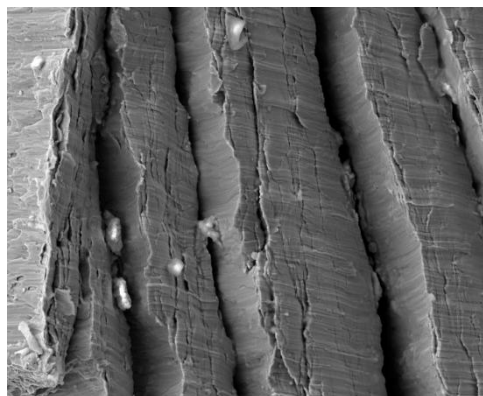


Optical Flow

Thomas Pock

$$\int_{\Omega} f(x, u(x), \nabla u(x)) dx \Leftrightarrow \sup_{\phi \in K} \int_{\Omega \times R} \phi \cdot D\mathbf{1}_u$$





Optical Flow (II)

- Content
 - Global approaches (Horn-Schunck, TV-L1)
 - Coarse-to-fine warping



The Horn and Schunck (HS) Method

Massachusetts Institute of Technology
Artificial Intelligence Laboratory

A. I. Memo No. 572

April 1980

Determining Optical Flow

Berthold K. P. Horn and Brian G. Schunck

Abstract. Optical flow cannot be computed locally, since only one independent measurement is available from the image sequence at a point, while the flow velocity has two components. A second constraint is needed. A method for finding the optical flow pattern is presented which assumes that the apparent velocity of the brightness pattern varies smoothly almost everywhere in the image. An iterative implementation is shown which successfully computes the optical flow for a number of synthetic image sequences. The algorithm is robust in that it can handle image sequences that are quantized rather coarsely in space and time. It is also insensitive to quantization of brightness levels and additive noise. Examples are included where the assumption of smoothness is violated at singular points or along lines in the image.

The Horn and Schunck (HS) Method

- Global energy to be minimized

$$E_{HS} = \|\nabla u\|_2^2 + \|\nabla v\|_2^2 + \lambda \|I_t + I_x(u - u_0) + I_y(v - v_0)\|_2^2$$

- λ is the regularization parameter
- The image is of size $w \times h$
- $u, v \in \mathbb{R}^{w \times h}$ are column vectors
- ∇ is a finite differences approximation of the gradient operator

$$(\nabla u)_{i,j} = \begin{pmatrix} \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < w \\ 0 & \text{else} \end{cases} \\ \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < h \\ 0 & \text{else} \end{cases} \end{pmatrix}$$

The linear system

- Optimality conditions:

$$\nabla^T \nabla u + \lambda I_x (I_t + I_x (u - u_0) + I_y (v - v_0)) = 0$$

$$\nabla^T \nabla v + \lambda I_y (I_t + I_x (u - u_0) + I_y (v - v_0)) = 0$$

- The optimality condition can be re-arranged as the following linear system

$$\begin{pmatrix} -\Delta + \lambda \text{diag}(I_x^2) & \lambda \text{diag}(I_x I_y) \\ \lambda \text{diag}(I_x I_y) & -\Delta + \lambda \text{diag}(I_y^2) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} I_x (I_x u_0 + I_y v_0 - I_t) \\ I_y (I_x u_0 + I_y v_0 - I_t) \end{pmatrix}$$

- System is large $2wh \times 2wh$ but very sparse
- Suitable solvers are Gauss-Seidel, CG, or Matlab “\”



The Horn and Schunck (HS) Method

- Advantages
 - Easy and fast to solve due to quadratic functions
 - Easy to implement
- Disadvantages
 - Quadratic smoothness term does not allow for sharp discontinuities in the motion field
 - Quadratic data term does not allow for outliers in the optical flow constraint



The TV- L^1 approach

A Duality Based Approach for Realtime TV- L^1 Optical Flow

C. Zach¹, T. Pock², and H. Bischof²

¹ VRVis Research Center

² Institute for Computer Graphics and Vision, TU Graz

Abstract. Variational methods are among the most successful approaches to calculate the optical flow between two image frames. A particularly appealing formulation is based on total variation (TV) regularization and the robust L^1 norm in the data fidelity term. This formulation can preserve discontinuities in the flow field and offers an increased robustness against illumination changes, occlusions and noise. In this work we present a novel approach to solve the TV- L^1 formulation. Our method results in a very efficient numerical scheme, which is based on a dual formulation of the TV energy and employs an efficient point-wise thresholding step. Additionally, our approach can be accelerated by modern graphics processing units. We demonstrate the real-time performance (30 fps) of our approach for video inputs at a resolution of 320×240 pixels.

1 Introduction

The recovery of motion from images is a major task of biological and artificial vision systems. The main objective of optical flow methods is to compute a flow field estimating the motion of pixels in two consecutive image frames. Since optical flow is an highly ill posed inverse problem, using pure intensity based



The TV-L1 approach

- Due to the quadratic functions, the HS method does not allow for discontinuities in the flow field
- A good idea is to replacing the quadratic functions by ℓ_1 norms
- Leads to the so-called TV-L1 approach

$$E_{TV-L1} = \|\nabla(u, v)\|_{2,1} + \lambda \|I_t + I_x(u - u_0) + I_y(v - v_0)\|_1$$

$$\|\nabla(u, v)\|_{2,1} = \sum_{i,j} \sqrt{|(\nabla u)_{i,j}|_2^2 + |(\nabla v)_{i,j}|_2^2}$$

- The first term is the so-called total variation of the flow field, the second term is the ℓ_1 norm of the OFC



Minimizing the TV-L1 energy

- The TV-L1 energy is convex but non-differentiable
- Standard gradient descent cannot be applied
- We can rely on recent advances in convex optimization
- We can apply two strategies:
 - Smooth the total variation term and apply the FISTA algorithm
 - Compute a saddle-point formulation of the energy and apply a primal-dual algorithm



The **FISTA** Algorithm

- **F**ast **I**terative **S**hrinkage **T**hresholding **A**lgorithm
- Proposed in 2008 by Beck and Teboulle
- **Can be applied to the following class of convex optimization problems**

$$\min_x f(x) + g(x)$$

- The function $f(x)$ has a Lipschitz continuous gradient

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

- The function $g(x)$ can be non-smooth but has a simple to compute proximal operator

$$p_L(y) = \arg \min_x \frac{L}{2} \|x - (y - \frac{1}{L} \nabla f(y))\|_2^2 + g(x)$$



FISTA

FISTA with constant stepsize

Input: $L = L(f)$ - A Lipschitz constant of ∇f .

Step 0. Take $\mathbf{y}_1 = \mathbf{x}_0 \in \mathbb{R}^n$, $t_1 = 1$.

Step k. ($k \geq 1$) Compute

$$(4.1) \quad \mathbf{x}_k = p_L(\mathbf{y}_k),$$

$$(4.2) \quad t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2},$$

$$(4.3) \quad \mathbf{y}_{k+1} = \mathbf{x}_k + \left(\frac{t_k - 1}{t_{k+1}} \right) (\mathbf{x}_k - \mathbf{x}_{k-1}).$$

Convergence rate: $(f + g)(x_k) - (f + g)(x^*) \leq \mathcal{O}(1/k^2)$



Application to the TV-L1 energy (1)

- Smoothing of the TV term to make its gradient Lipschitz

$$f(u, v) \equiv \|\nabla(u, v)\|_\varepsilon = \sum_{i,j} h_\varepsilon(|(\nabla u)_{i,j}|_2^2 + |(\nabla v)_{i,j}|_2^2)$$

- where $h_\varepsilon(t^2)$ denotes the Huber function, $\varepsilon > 0$

$$h_\varepsilon(t^2) = \begin{cases} \frac{t^2}{2\varepsilon} + \frac{\varepsilon}{2} & \text{if } \sqrt{t^2} \leq \varepsilon \\ \sqrt{t^2} & \text{else} \end{cases}$$

- $f(u, v)$ has a Lipschitz continuous gradient

$$\nabla f(u, v) = \nabla^T \text{diag}\left(\frac{1}{\max(\varepsilon, |\nabla(u, v)|)}\right) \nabla(u, v)$$

- with $L(f) = \|\nabla\|^2/\varepsilon = 8/\varepsilon$

Application to the TV-L1 energy (2)

- The non-smooth function $g(u, v)$ is given by the ℓ_1 norm of the data term

$$g(u, v) \equiv \lambda \|I_t + I_x(u - u_0) + I_y(v - v_0)\|_1$$

- The solution of the proximal operator is given by

$$p_L(u, v) = (\hat{u}, \hat{v}) + \begin{cases} +\frac{\lambda}{L} \nabla I & \text{if } \rho(\hat{u}, \hat{v}) < -\frac{\lambda}{L} |\nabla I|_2^2 \\ -\frac{\lambda}{L} \nabla I & \text{if } \rho(\hat{u}, \hat{v}) > +\frac{\lambda}{L} |\nabla I|_2^2 \\ -\frac{\rho(\hat{u}, \hat{v})}{|\nabla I|_2^2} \nabla I & \text{else} \end{cases}$$

- where $(\hat{u}, \hat{v}) = (u, v) - \frac{1}{L} \nabla f(u, v)$
- and $\rho(\hat{u}, \hat{v}) = I_t + I_x(\hat{u} - u_0) + I_y(\hat{v} - v_0)$



Primal-dual optimization

- A first-order primal-dual algorithm proposed in Chambolle, Pock, 2011
- Can be used to find a saddle point of the following class of convex-concave saddle-point problems

$$\min_x \max_y \langle Kx, y \rangle + G(x) - F^*(y)$$

- where K is a linear operator, G and F^* are convex (non-smooth) functions and have simple proximal operators
- Corresponds to a saddle-point formulation of the primal and dual problems

$$\min_x F(Kx) + G(x) \qquad \max_y -(F^*(y) + G(-K^T y))$$

The algorithm

- Initialization: Choose $\tau, \sigma > 0$, $\theta \in [0, 1]$, $(x^0, y^0) \in X \times Y$ and set $\bar{x}^0 = x^0$.
- Iterations ($n \geq 0$): Update x^n, y^n, \bar{x}^n as follows:

$$\begin{cases} y^{n+1} = (I + \sigma \partial F^*)^{-1}(y^n + \sigma K \bar{x}^n) \\ x^{n+1} = (I + \tau \partial G)^{-1}(x^n - \tau K^* y^{n+1}) \\ \bar{x}^{n+1} = x^{n+1} + \theta(x^{n+1} - x^n) \end{cases} \quad (7)$$

- Step sizes: $\tau \sigma \|K\|^2 < 1$, $\theta = 1$
- Proximal operators

$$(I + \tau \partial G)^{-1}(\hat{x}) = \arg \min_x \frac{\|x - \hat{x}\|_2^2}{2\tau} + G(x)$$

$$(I + \sigma \partial F^*)^{-1}(\hat{y}) = \arg \min_y \frac{\|y - \hat{y}\|_2^2}{2\sigma} + F^*(y)$$



Saddle-point formulation of TV-L1

- The total variation can be written as (convex conjugate)

$$\|\nabla(u, v)\|_{2,1} = \max_{\|p\|_{2,\infty} \leq 1} \langle \nabla(u, v), p \rangle$$

- The TV-L1 optical flow model is written as

$$\min_{u,v} \max_p \langle \nabla(u, v), p \rangle + G(u, v) - F^*(p)$$

- with $G(u, v) \equiv \lambda \|I_t + I_x(u - u_0) + I_y(v - v_0)\|_1$

$$F^*(p) \equiv \begin{cases} 0 & \text{if } \|p\|_{2,\infty} \leq 1 \\ \infty & \text{else} \end{cases}$$

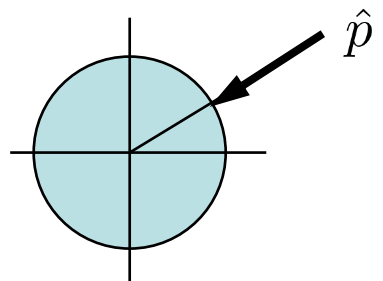
- Exactly falls into the class of the primal-dual algorithm
- $x \equiv (u, v)$, $y \equiv p$, $K \equiv \nabla$

The proximal operators

- The proximal operator with respect to G is given by the same soft-shrinkage formula as before

$$(I + \tau \partial G)^{-1}(\hat{u}, \hat{v}) = (\hat{u}, \hat{v}) + \begin{cases} +\tau \lambda \nabla I & \text{if } \rho(\hat{u}, \hat{v}) < -\tau \lambda |\nabla I|_2^2 \\ -\tau \lambda \nabla I & \text{if } \rho(\hat{u}, \hat{v}) > +\tau \lambda |\nabla I|_2^2 \\ -\frac{\rho(\hat{u}, \hat{v})}{|\nabla I|_2^2} \nabla I & \text{else} \end{cases}$$

- The proximal operator with respect to F^* is a projection onto the Euclidean ball

$$(I + \sigma \partial F^*)^{-1}(\hat{p}) = \frac{\hat{p}}{\max(1, |\hat{p}|_2)}$$




The TV-L1 approach

- Advantages
 - TV regularization allows for discontinuities in the flow field
 - The L1 data term allows for outliers (occlusions)
 - Reasonable fast to compute (GPU leads to realtime)
- Disadvantages
 - The method requires an iterative solver to compute the minimizer (FISTA, or Primal-Dual)
 - FISTA requires to smooth the TV
 - Primal-Dual can also deal with the pure TV
 - Hard to find a good stopping criterion

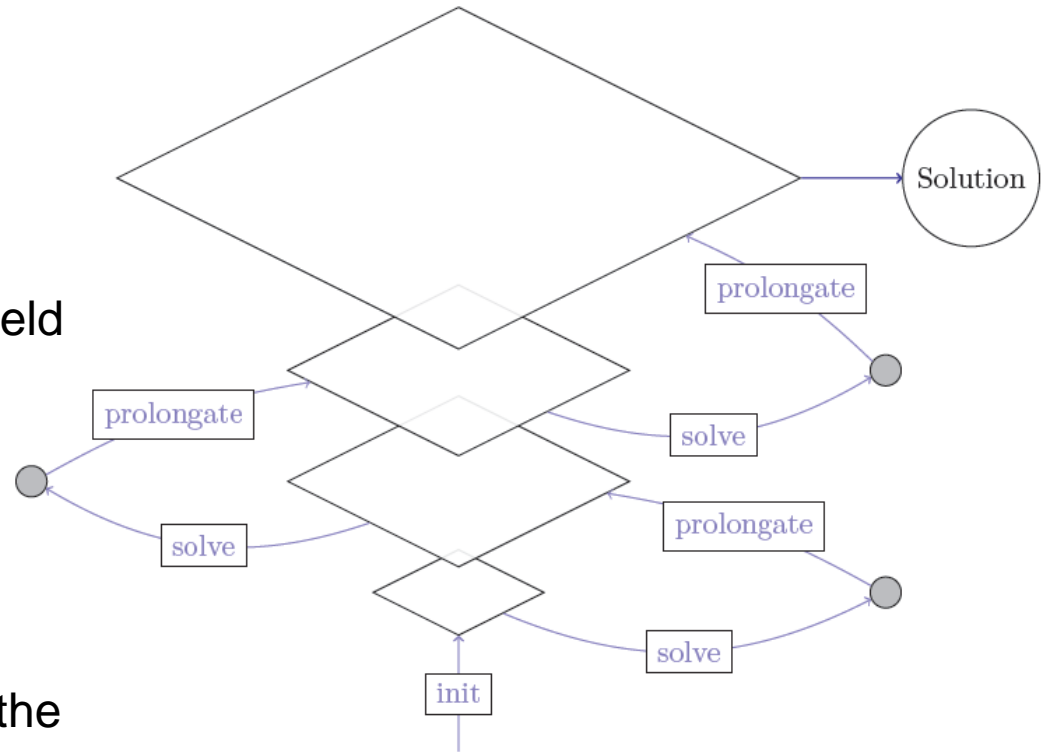
Coarse-to-fine warping framework

- Due to the restrictions of the OFC, the discussed methods are only able to recover small motion
- How to extend the method for large motion?
 1. Solving, warping, re-linearization, solving, ...
 2. Implement the method on an image pyramid



Coarse-to-fine warping framework

- **Compute image pyramids**
- **init**
 - Initialize the flow field
- **solve**
 - Transform the moving image by the given flow field (warping)
 - Perform linearization
 - Apply one of the three methods (LK, HS, TV-L1)
- **prolongate**
 - Initialize the flow field on the next finer level using interpolation and rescaling of the motion vectors





Coarse-to-fine warping framework

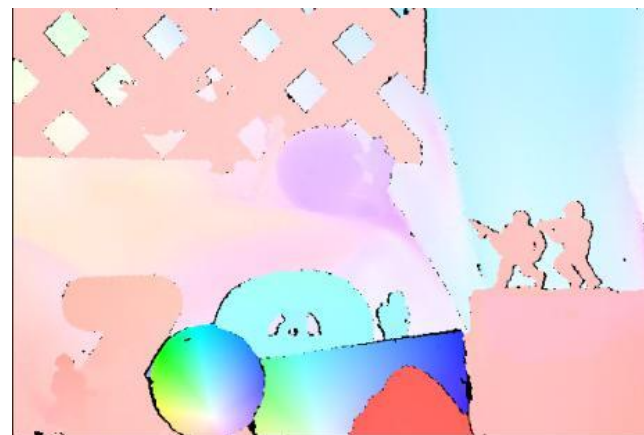
- Advantages
 - Allows to compute large motion
 - Due to the logarithmic nature of pyramids, not much more to compute
- Disadvantages
 - Large motion that is not captured at a coarse scale cannot be found on finer levels
 - Dilemma: Large motion of small objects
 - Different interpolation, rescaling, filtering schemes lead to different results
 - Can lead to unstable results in practice
- **BUT: What is the alternative?**

Comparison of the three methods

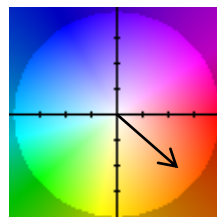
- In the following, we will show an comparison of the three methods we covered so far
- We use the “Army” sequence of the Middlebury benchmark



Input frames



Ground truth flow



(u, v)

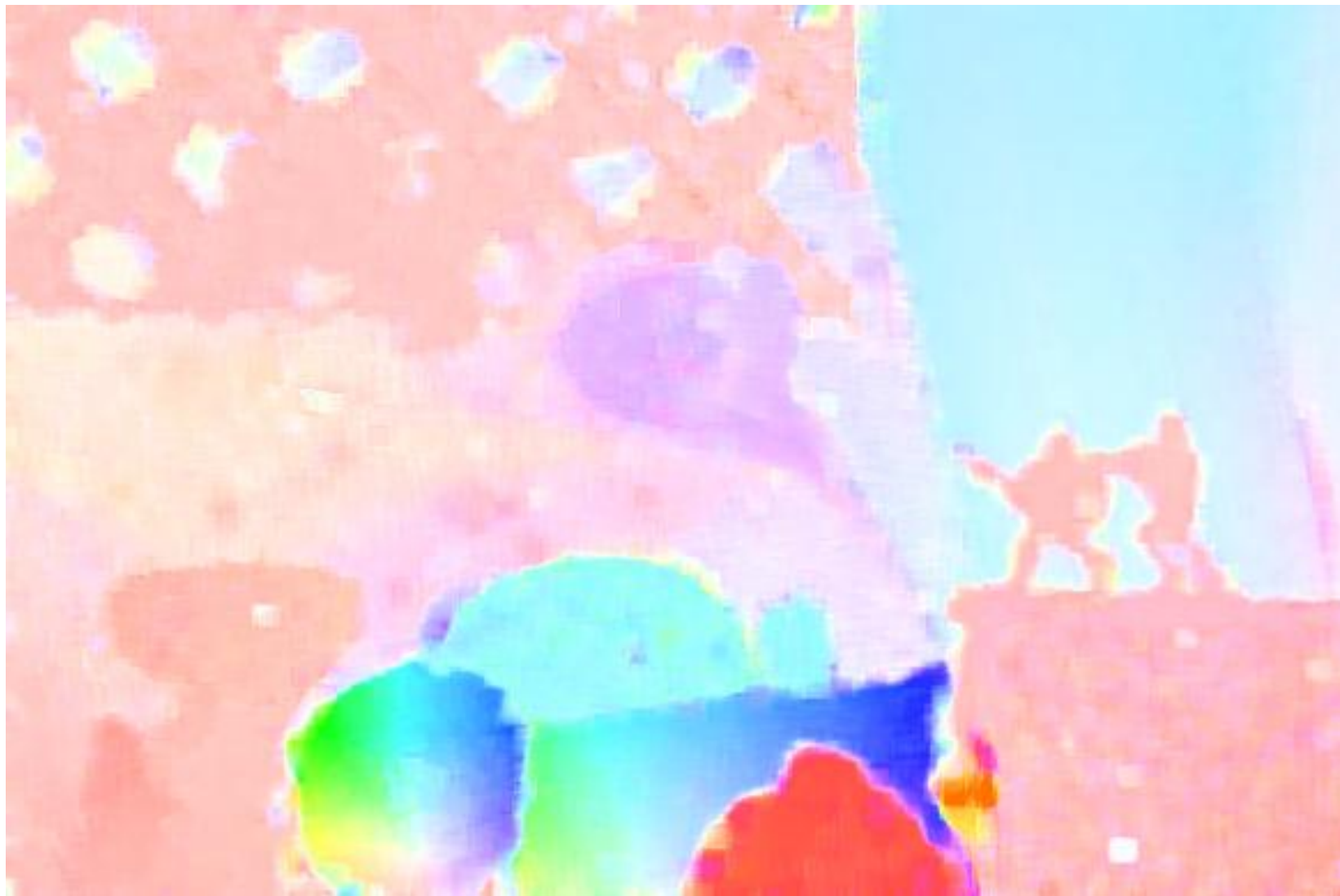
Structure-texture decomposition

- The data set contains changing illumination and shadows



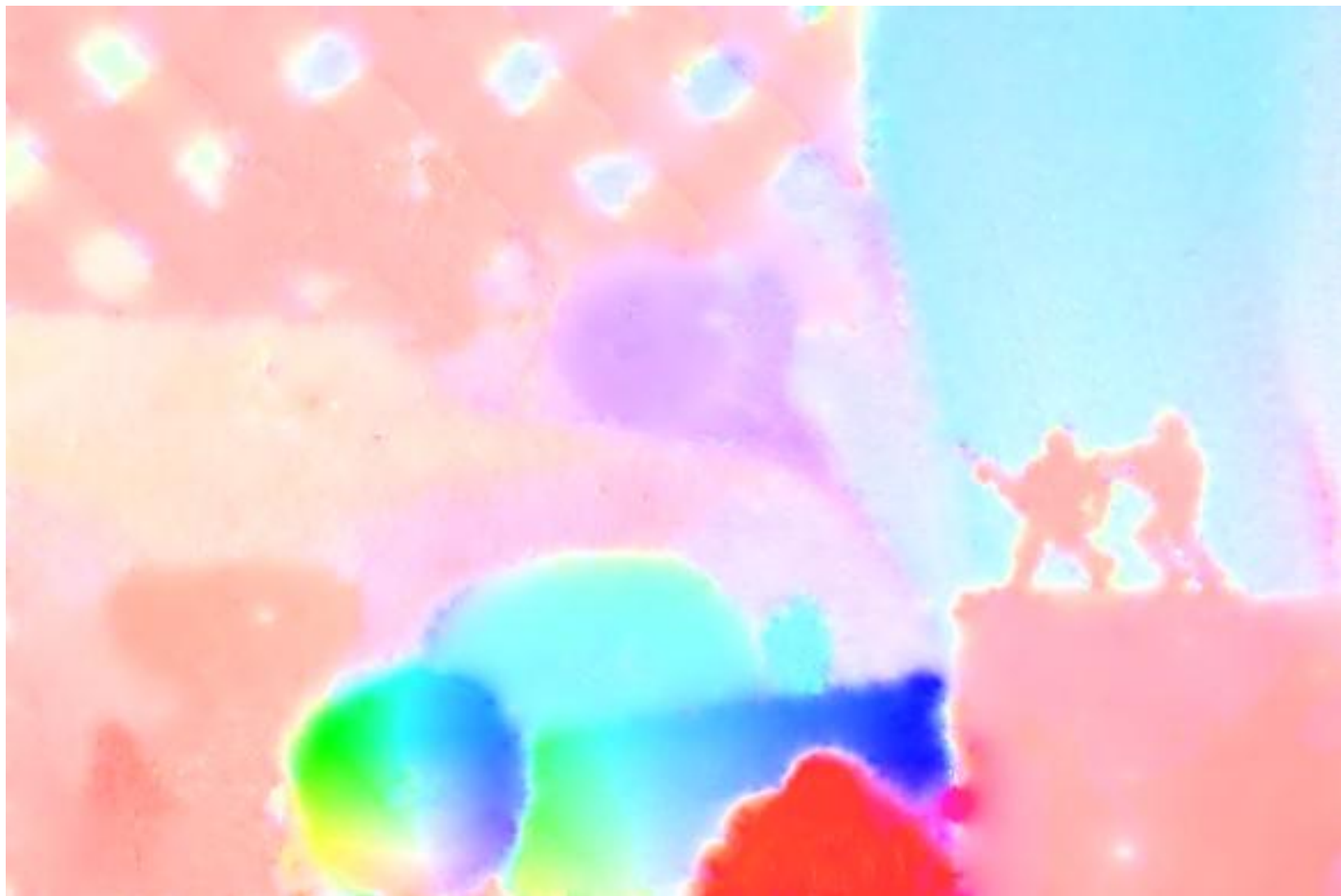


Lucas-Kanade



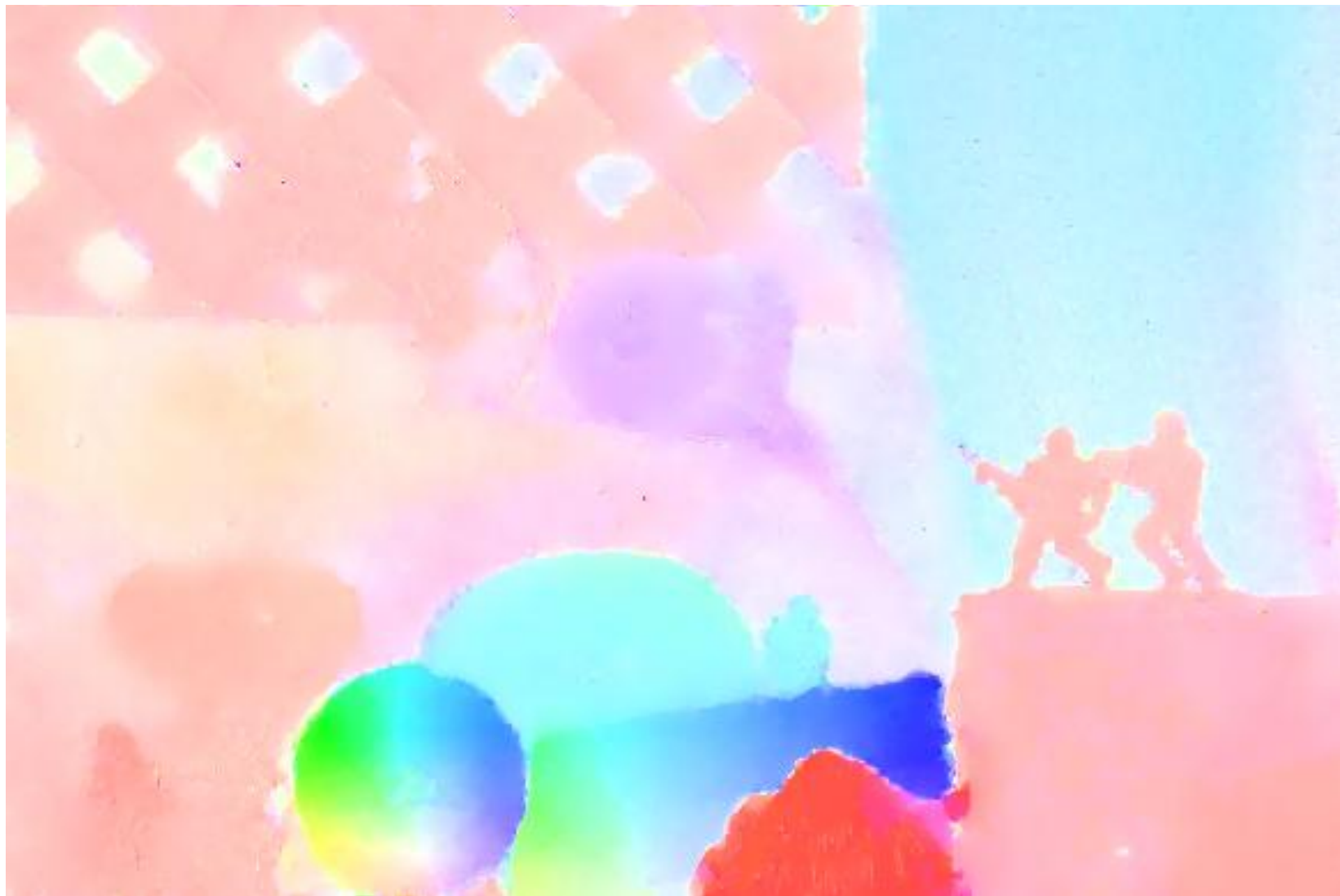


Horn-Schunck





TV-L1



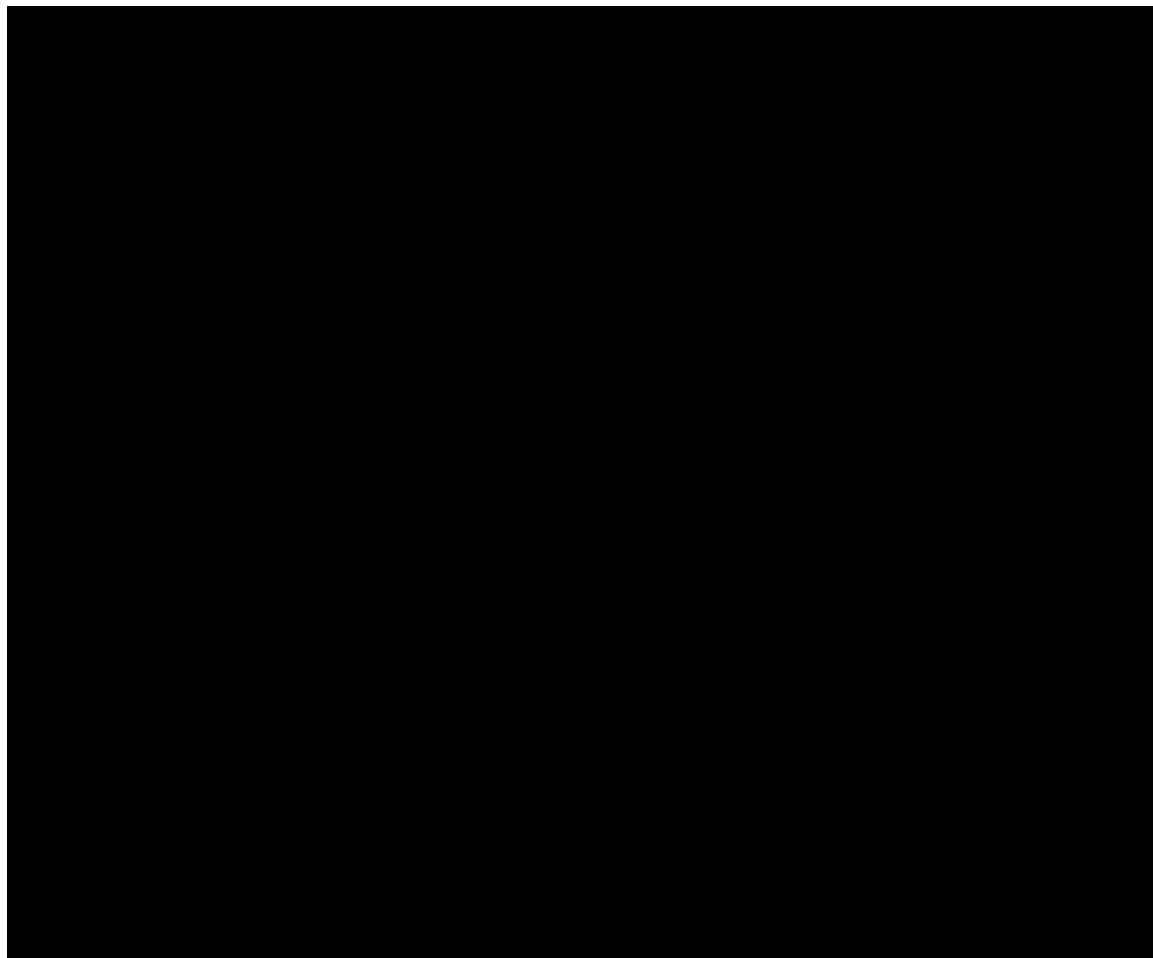
Real-time implementation

- Implementation of the TV-L1 method on a GPU allows for real-time computation





Flow Games



Jakob Santner et al.

Feature Flow

- How can we compute the motion between challenging sequences
 - Large displacements
 - Different modalities
 - Image taken at different time points
 - Images from objects of the same “category” 😊





Feature Flow

- The idea is simple
- Replace each pixel in both images by a feature vector
 - SIFT descriptor
 - LBP
 - ...
- Feature constancy assumption

$$\mathcal{F}(x + \Delta x, y + \Delta y, z + \Delta z, t + \Delta t) \approx \mathcal{F}(x, y, z, t)$$

- Linearize each feature channel individually



Some results using SIFT





Some results using SIFT





Some results using SIFT





Some results using SIFT





Some results using SIFT



Some results using SIFT

